ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΚΜΑΘΗΣΗ ΚΥΚΛΩΝ ΣΤΑ ΚΟΙΝΩΝΙΚΑ ΔΙΚΤΥΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΩΝΣΤΑΝΤΙΝΟΣ ΠΗΓΙΩΤΗΣ

Επιβλέπουσα: Θεοδώρα Βαρβαρίγου

Καθηγήτρια Ε.Μ.Π

Αθήνα, Απρίλιος 2016

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΚΜΑΘΗΣΗ ΚΥΚΛΩΝ ΣΤΑ ΚΟΙΝΩΝΙΚΑ ΔΙΚΤΥΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΩΝΣΤΑΝΤΙΝΟΣ ΠΗΓΙΩΤΗΣ

Επιβλέπουσα: Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την $15^{η}$ Απριλίου 2016.

.....	.....	.....
Θεοδώρα Βαρβαρίγου	Βασίλειος Λούμος	Εμμανουήλ Βαρβαρίγος
Καθηγήτρια Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π

Αθήνα, Απρίλιος 2016

.....

Κωνσταντίνος Πηγιώτης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός & Μηχανικός Υπολογιστών Ε.Μ.Π.

ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια η ανάπτυξη των κοινωνικών δικτύων είναι ραγδαία. Καθημερινά όλο και περισσότεροι άνθρωποι γίνονται μέλη σε πολλές διαδικτυακές πλατφόρμες. Κάποιες από τις πιο γνωστές αποτελούν το Facebook, Twitter και Google+. Οι χρήστες των κοινωνικών δικτύων πολλές φορές έχουν μερικές εκατοντάδες ίσως και χιλιάδες φίλους ή ακολούθους με αποτέλεσμα το προσωπικό τους δίκτυο να γίνεται αρκετά μεγάλο και περίπλοκο. Αυτό όμως οδηγεί στο εξής πρόβλημα, ότι οι χρήστες δεν μπορούν με εύκολο και αυτοματοποιημένο τρόπο να δημιουργούν ομάδες και να κατηγοριοποιούν τους φίλους τους και έτσι η συνολική εμπειρία που έχουν δεν είναι αυτή που ίσως επιθυμούν.

Μέχρι στιγμής οι ιστοσελίδες κοινωνικής δικτύωσης επιτρέπουν στους χρήστες να κατηγοριοποιούν τους φίλους τους χειροκίνητα. Παρόλα αυτά, η διαδικασία είναι αρκετά δύσκολη και χρονοβόρα και απαιτεί ανανέωση κάθε φορά που αλλάζει το προσωπικό δίκτυο κάθε χρήστη.

Στην παρούσα διπλωματική εργασία μελετώνται διάφορες τεχνικές μη επιβλεπόμενης μηχανικής μάθησης που θα μπορούσαν να χρησιμοποιηθούν για μοντελοποίηση του εξής προβλήματος. Ο αλγόριθμος ο οποίος κατασκευάσαμε προσπαθεί να ομαδοποιήσει τους χρήστες κάποιου κοινωνικού δικτύου με βάση τις σχέσεις που υπάρχουν μεταξύ τους και επίσης με βάση κάποια κοινά χαρακτηριστικά από το προφίλ κάθε χρήστη. Τέλος ο αλγόριθμος μας μπορεί να κατασκευάσει κύκλους οι οποίο αλληλεπικαλύπτονται καθώς και ένθετους κύκλους.

Λέξεις κλειδιά: μηχανική μάθηση, μη-επιβλεπόμενη, ομαδοποίηση, κοινωνικά δίκτυα, εκμάθηση κύκλων, εκμάθηση κοινοτήτων

ABSTRACT

In recent years the development of social networks is rapidly increasing. Everyday more and more people join in many internet platforms. Some of the best known are the Facebook, Twitter and Google+, among others. Users of social networks often have hundreds and perhaps thousands of friends or followers and subsequently, their personal network can become quite large and complex. But this leads to the following problem that users can not in an easy and automated way create groups and categorize their friends and thus, overall experience they have is not what they wish.

So far social networking sites allow users to categorize their friends manually. However, the process is quite difficult and time consuming and requires renewal every time a user change their private network.

This thesis has studied various non-supervised machine learning techniques that could be used for modeling this specific problem. The algorithm that we constructed is trying to group users of a social network based on the relationships that exist between them and also based on some common features of the profile of each user. Finally, our algorithm can construct circles that overlap and also nested circles.

Keywords: machine learning, non-supervised, clustering, social networks, learning circles, learning communities

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα αρχικά να ευχαριστήσω την Καθηγήτρια Ε.Μ.Π κυρία Θεοδώρα Βαρβαρίγου για την εμπιστοσύνη που μου έδειξε δίνοντας μου την ευκαιρία να ασχοληθώ με ένα τόσο επίκαιρο και ενδιαφέρον θέμα. Επίσης θα ήθελα να ευχαριστήσω όλα τα μέλη της πανεπιστημιακής κοινότητας με τα οποία συναναστράφηκα αυτά τα χρόνια και με βοήθησαν να φέρω εις πέρας τις σπουδές μου. Θα ήθελα να ευχαριστήσω τους καθηγητές μου, που όλα αυτά τα χρόνια με εφοδίασαν με τις απαραίτητες γνώσεις έτσι ώστε να πετύχω στη καριέρα μου και γενικότερα στη ζωή μου. Θα ήθελα επίσης να ευχαριστήσω την οικογένεια μου που ήταν πάντα δίπλα μου σε όλες τις δυσκολίες που αντιμετώπισα όλα αυτά τα χρόνια στο Ε.Μ.Π. Τέλος θα ήθελα να ευχαριστήσω όλους τους φίλους και φίλες που γνώρισα σε αυτό το ταξίδι και τους εύχομαι με τη σειρά μου, να έχουν κάθε επιτυχία σε όλους τους τομείς της ζωής τους.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Μέρος Ι

ΕΙΣΑΓΩΓΗ

1

ΣΚΟΠΟΣ ΤΗΣ ΠΑΡΟΥΣΑΣ ΔΗΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Τα μέσα κοινωνικής δικτύωσης χρησιμοποιούνται καθημερινά από εκατομμύρια χρήστες ανά το παγκόσμιο. Εν έτη 2016 υπάρχουν δεκάδες μέσα κοινωνικής δικτύωσης, καθένα από τα οποία έχει συγκεκριμένες δυνατότητες και λειτουργίες προσελκύοντας με αυτό το τρόπο διαφορετικό ακροατήριο. Ο κοινός παρονομαστής σχεδόν σε όλα τα κοινωνικά δίκτυα είναι η δημιουργία δικτύων ανθρώπων που έχουν κάποια κοινά χαρακτηριστικά όπως κοινά ενδιαφέροντα, κοινές δραστηριότητες και όμοιες απόψεις.

Έχουμε παρατηρήσει διάφορα συστήματα μετάδοσης πληροφορίας στο διαδίκτυο, μεταξύ των οποίων τοποθετούμε και τα κοινωνικά δίκτυα. Σε αντίθεση όμως με οτιδήποτε άλλο υπάρχει στο διαδίκτυο, τα κοινωνικά δίκτυα περιστρέφονται γύρω από το χρήστη και όχι από το περιεχόμενο. Οι χρήστες που είναι ενεργά μέλη σε ένα κοινωνικό δίκτυο, δημιουργούν συνδέσμους με άλλους χρήστες και δημοσιεύουν το προφίλ τους με περιεχόμενο που περικλείεται σε αυτό. Αυτό έχει ως αποτέλεσμα τη δημιουργία κοινωνικών σχέσεων, την εύρεση χρηστών με κοινά ενδιαφέροντα και τον εντοπισμό γνώσης που παρέχεται από τα μέλη του δικτύου.

Τα κοινωνικά δίκτυα μας επιτρέπουν να ακολουθούμε ροές πληροφοριών που δημιουργούνται από εκατοντάδες φίλους και γνωστούς. Τα άτομα που ακολουθούμε παράγουν τεράστιο όγκο πληροφοριών και για να αντιμετωπίσουμε τον υπερβολικό φόρτο πληροφοριών πρέπει με κάποιο τρόπο να τα οργανώσουμε. Ένας από τους κύριους μηχανισμούς για τους χρήστες των υπηρεσιών κοινωνικής δικτύωσης για να οργανώσουν τα δίκτυά τους και το περιεχόμενο που παράγεται από αυτά, είναι να κατηγοριοποιήσουν τους φίλους τους, σε κοινωνικούς κύκλους.

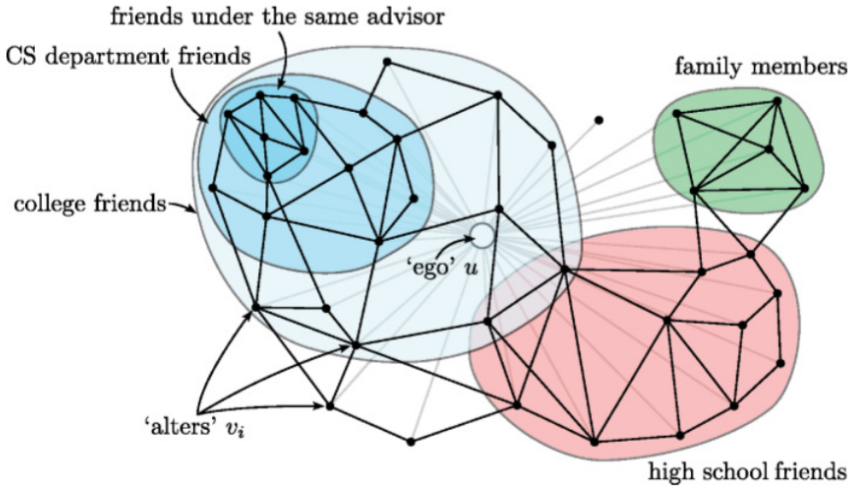
Οι κοινωνικοί κύκλοι αποτελούν λίστες ατόμων που μπορεί να χρησιμοποιηθούν για το φιλτράρισμα του περιεχομένου που προβάλλεται, για την προστασία της ιδιωτικής ζωής, καθώς και για την ανταλλαγή ομάδων χρηστών που οι υπόλοιποι μπορεί να επιθυμούν να ακολουθήσουν. Σχεδόν όλα τα μεγάλα κοινωνικά δίκτυα παρέχουν αυτή τη λειτουργικότητα. Για παράδειγμα, κοινωνικοί κύκλοι, στο Google+ ονομάζονται απλά κύκλοι, στο Twitter και στο Facebook ονομάζονται λίστες.

Ο "ιδιοκτήτης", που στη περίπτωση μας είναι ο χρήστης, ενός τέτοιου δικτύου μπορεί να σχηματίσει κύκλους με βάση τα κοινά χαρακτηριστικά και δεσμούς που έχουν οι χρήστες που ακολουθεί. Για παράδειγμα, ο χρήστης μπορεί να επιθυμεί να μοιραστεί, το τελευταίο άρθρο του theverge.com μόνο με τους φίλους τους από το τμήμα πληροφορικής, ενώ τις οικογενειακές φωτογραφίες μόνο με το στενό του οικογενειακό κύκλο.

Αντίστοιχα, μπορεί να επιθυμεί να περιορίσει την ποσότητα των πληροφοριών που λαμβάνει από τους παιδικούς του φίλους. Για αυτό οι κύκλοι θεωρούνται αναπόσπαστο κομμάτι των κοινωνικών δικτύων, και αυτές είναι οι λειτουργίες που επιδιώκουν να διευκολύνουν.

Αυτή τη στιγμή, οι χρήστες σε Facebook, Google+ και Twitter προσδιορίζουν τους κύκλους τους, είτε χειροκίνητα είτε με κάποιο αφελή τρόπο, για παράδειγμα, με τον εντοπισμό τους φίλους τους που έχουν ορισμένα κοινά χαρακτηριστικά ή ιδιότητες. Καμία προσέγγιση δεν είναι ιδιαίτερα ικανοποιητική. Η πρώτη είναι χρονοβόρα και δεν ενημερώνεται αυτόματα καθώς ο χρήστης προσθέτει περισσότερους χρήστες στο κοινωνικό του δίκτυο, ενώ η δεύτερη αποτυγχάνει αρκετά συχνά κυρίως όταν κάποια από τα στοιχεία ενός χρήστη δεν υπάρχουν ή δεν είναι ακριβή. Οι κύκλοι αποτελούν χαρακτηριστικό κάθε συγκεκριμένου χρήστη, καθώς κάθε χρήστης οργανώνει το προσωπικό του δίκτυο, με διαφορετικό τρόπο από όλους τους άλλους χρήστες. Οι κύκλοι μπορεί να επικαλύπτονται (ένας κύκλος φίλων από την ίδια πόλη μπορεί να επικαλύπτεται με έναν κύκλο φίλων από το ίδιο το πανεπιστήμιο).[42]

Παραδείγματα κύκλων που δημιουργούνται από ένα χρήστη παρουσιάζονται στην πιο κάτω εικόνα.



Σχήμα 1.1: Ένα δίκτυο με επισημασμένους κύκλους. Ο κεντρικός χρήστης (u), είναι φίλος με όλους τους άλλους χρήστες στο δίκτυο.

Παρά το γεγονός ότι, ένας κύκλος, δεν είναι ακριβώς το ίδιο με μια κοινότητα, οι βασικές αρχές ανίχνευσης κοινοτήτων ισχύουν σε μεγάλο βαθμό.[38] Ενώ οι κλασικοί αλγόριθμοι ομαδοποίησης υποθέτουν ασύνδετες κοινότητες πολλοί ερευνητές έχουν προβεί στη παρατήρηση ότι οι κοινότητες στο πραγματικό κόσμο μπορούν να επικαλύπτονται ή να έχουν ιεραρχική δομή.[47]

Σε αυτή τη διπλωματική εργασία θα ασχοληθούμε με το πρόβλημα της αυτόματης δημιουργίας κοινωνικών κύκλων του κάθε χρήστη. Συγκεκρι-

μένα, δίνεται ένας χρήστης με το προσωπικό του κοινωνικό δίκτυο και ο στόχος μας είναι να δημιουργήσουμε τους κύκλους του, καθένας από τους οποίους αποτελεί ένα υποσύνολο των φίλων του. Για να επιτευχθεί, η αυτοματοποιημένη εισαγωγή των φίλων του χρήστη σε κοινωνικούς κύκλους, θα χρησιμοποιήσουμε διάφορες τεχνικές μη επιβλεπόμενης μηχανικής μάθησης.

Οι κύκλοι είναι διαφορετικοί για κάθε χρήστη, καθώς κάθε χρήστης οργανώνει το προσωπικό του δίκτυο διαφορετικά από κάθε άλλο χρήστη που είναι συνδεδεμένος μαζί του. Αυτό σημαίνει ότι μπορούμε να ορίσουμε το πρόβλημα της ανίχνευσης κύκλων ως ένα πρόβλημα ομαδοποίησης. Στη πράξη οι κύκλοι μπορεί να αλληλοκαλύπτονται μεταξύ τους ή να οργανώνονται ιεραρχικά (για παράδειγμα οι χρήστες από ένα μάθημα στο πανεπιστήμιο είναι υποσύνολο του δικτύου των χρηστών που φοιτούν σε αυτό το πανεπιστήμιο). Σχεδιάσαμε το μοντέλο μας, έχοντας υπόψιν και τους δύο τρόπους συμπεριφοράς.

Σε γενικές γραμμές, υπάρχουν δύο χρήσιμες πηγές δεδομένων που θα βοηθήσουν στη σχεδίαση του αλγόριθμου. Η πρώτη, είναι το σύνολο των ακμών του δικτύου. Αναμένουμε ότι, οι κύκλοι σχηματίζονται από πυκνά συνδεδεμένα σύνολα από φίλους.[43]Ωστόσο, οι κύκλοι επικαλύπτονται σε μεγάλο βαθμό δηλαδή οι φίλοι του χρήστη ανήκουν σε πολλαπλούς κύκλους ταυτόχρονα. Επίσης, πολλοί κύκλοι είναι ιεραρχικά τοποθετημένοι μέσα σε μεγαλύτερους. Έτσι, είναι σημαντικό να μοντελοποιήσουμε την ιδιότητα των χρηστών να είναι μέλη σε πολλούς κύκλους ταυτόχρονα. Δεύτερον, αναμένουμε ότι κάθε κύκλος είναι όχι μόνο πυκνά συνδεδεμένος, αλλά ότι τα μέλη του έχουν επίσης κοινά χαρακτηριστικά ή ιδιότητες.[25]Ως εκ τούτου, θα πρέπει να μοντελοποιήσουμε ρητά τις διάφορες συνιστώσες των προφίλ των χρηστών για τους κύκλους που προκύπτουν.

Θα μοντελοποιήσουμε τις ομοιότητες μεταξύ χρηστών ως μια συνάρτηση κοινών χαρακτηριστικών των προφίλ των χρηστών. Προτείνουμε, μια μη επιβλεπόμενη μέθοδο μηχανικής μάθησης για να μάθουμε ποια χαρακτηριστικά των προφίλ των χρηστών, οδηγούν σε πυκνά συνδεδεμένους κύκλους.

Το μοντέλο μας έχει δύο βασικές διαφορές σε σύγκριση με τα υπάρχον μοντέλα. Κατ' αρχάς, κάθε κόμβος είτε αποτελεί μέλος του κύκλου είτε όχι. Δεύτερον, προτείνοντας ένα παραμετροποιημένο ορισμό της ομοιότητας των προφίλ μαθαίνουμε ποια χαρακτηριστικά είναι σημαντικότερα για τη δημιουργία κάθε κύκλου. Αυτό το πετυχαίνουμε επιτρέποντας σε κάθε κύκλο να έχει ένα διαφορετικό ορισμό για την ομοιότητα των προφίλ των χρηστών. Δηλαδή, θα μπορούσαν να δημιουργηθούν κύκλοι με βάση το ίδιο σχολείο ή με βάση τη τοποθεσία που μένει ο χρήστης. Όπως παρατηρούμε (Σχήμα 1), οι δύο κύκλοι δημιουργήθηκαν από διαφορετικά χαρακτηριστικά (π.χ Σχολείο, Τόπος Διαμονής).

Αξιολογούμε τη μέθοδο μας σε σύνολο δεδομένων 1143 δικτύων από τα κοινωνικά δίκτυα Facebook, Twitter και Google+, για τα οποία γνω-

ρίζουμε ήδη από πριν ότι περιέχουν 5636 κύκλους, τους οποίους έχουν δημιουργήσει οι χρήστες χειροκίνητα.

Τα πειραματικά αποτελέσματα δείχνουν ότι, λαμβάνοντας ταυτόχρονα υπόψη τη δομή των κοινωνικών δικτύων, καθώς και πληροφορίες για το προφίλ του χρήστη, η μέθοδός μας αποδίδει σημαντικά καλύτερα από τις φυσικές εναλλακτικές. Πέραν του ότι είναι πιο ακριβής, η μέθοδος μας επιτρέπει επίσης να εξηγήσουμε γιατί ορισμένοι κόμβοι ανήκουν σε κοινές κοινότητες. Η μέθοδος μας υλοποιεί αλγόριθμο μηχανικής μάθησης χωρίς επίβλεψη και είναι σε θέση να προσδιορίσει αυτόματα τόσο τον αριθμό των κύκλων, καθώς και τους ίδιους τους κύκλους.

Μέρος II

ΚΟΙΝΩΝΙΚΑ ΔΙΚΤΥΑ

2

ΠΑΡΑΓΩΓΙΚΟ ΜΟΝΤΕΛΟ ΓΙΑ ΦΙΛΙΕΣ ΣΤΑ ΚΟΙΝΩΝΙΚΑ ΔΙΚΤΥΑ

2.1 Κοινωνικά δίκτυα

Κοινωνικά δίκτυα ονομάζουμε τις διαδικτυακές τεχνολογικές πλατφόρμες οι οποίες επιτρέπουν τη δημοσίευση περιεχομένου και πληροφοριών, την ανάπτυξη κατ' αρχήν επικοινωνιακών και στη συνέχεια κοινωνικών, πολιτικών και οικονομικών σχέσεων και δεσμών μεταξύ των ανθρώπων.

Μια υπηρεσία κοινωνικής δικτύωσης αποτελεί μια πλατφόρμα που έχει ως σκοπό την οικοδόμηση κοινωνικών σχέσεων μεταξύ ανθρώπων που έχουν κοινά ενδιαφέροντα, δραστηριότητες, υπόβαθρα ή συνδέσεις στη πραγματική ζωή. Αποτελείται από την απεικόνισή του κάθε χρήστη, τις συνδέσεις που υπάρχουν μεταξύ τους και από μια πληθώρα επιπρόσθετων υπηρεσιών. Τα περισσότερα κοινωνικά δίκτυα είναι βασισμένα σε web based υπηρεσίες που επιτρέπουν στους χρήστες τους να δημιουργήσουν ένα δημόσιο προφίλ, μία λίστα με άλλους χρήστες οι οποίοι αποτελούν τις συνδέσεις τους, καθώς και να βλέπουν τις συγκεκριμένες συνδέσεις. Παρέχουν στους χρήστες, τη δυνατότητα να αλληλεπιδρούν μέσω του διαδικτύου με ηλεκτρονική αλληλογραφία, μηνύματά και αρκετές άλλες υπηρεσίες ανάλογα με το είδος του κοινωνικού δικτύου. Τα μέσα κοινωνικής δικτύωσης επιτρέπουν επίσης στους χρήστες να μοιράζονται ιδέες, εικόνες, δραστηριότητες, εκδηλώσεις και ενδιαφέροντα.[1]

Με βάση την ιδέα ότι κάθε δύο άνθρωποι στον πλανήτη θα μπορούσαν να έρθουν σε επαφή μέσω μιας αλυσίδας που δεν υπερβαίνει τους πέντε ενδιάμεσους χρήστες, η κοινωνική δικτύωση θεσπίζει διασυνδεδεμένες διαδικτυακές κοινότητες (μερικές φορές γνωστές και ως κοινωνικά γραφήματα) που βοηθούν τους ανθρώπους να κάνουν διαφορετικά με βάση κάποια κοινά ενδιαφέροντα, που θα ήταν αδύνατο να γνωριστούν διαφορετικά.[10]

2.1.1 *Facebook*

Το Facebook είναι ιστοχώρος κοινωνικής δικτύωσης που ιδρύθηκε στις 4 Φεβρουαρίου του 2004. Οι χρήστες μπορούν να επικοινωνούν μέσω μηνυμάτων με τις επαφές τους και να τους ειδοποιούν όταν ανανεώνουν τις προσωπικές πληροφορίες τους. Όλοι έχουν ελεύθερη πρόσβαση στο να συμμετάσχουν σε δίκτυα που σχετίζονται μέσω πανεπιστημίου, θέσεων απασχόλησης ή γεωγραφικών περιοχών. Η εγγραφή είναι δωρεάν, και όπως έχει δηλώσει ο δημιουργός του 'είναι δωρεάν και θα είναι για πάντα'.

Ο Μαρκ Ζάκερμπεργκ ίδρυσε το Facebook ως μέλος του Πανεπιστημίου Χάρβαρντ. Αρχικά δικαίωμα συμμετοχής είχαν μόνο οι φοιτητές του Χάρβαρντ ενώ αργότερα επεκτάθηκε για το Ivy League. Το όνομα της ιστοσελίδας προέρχεται από τα έγγραφα παρουσίασης των μελών πανεπιστημιακών κοινοτήτων μερικών Αμερικάνικων κολεγίων που χρησιμοποιούσαν οι νεοεισερχόμενοι σπουδαστές για να γνωριστούν μεταξύ τους. Το 2005 το δικαίωμα πρόσβασης επεκτάθηκε σε μαθητές συγκεκριμένων λυκείων και μέλη ορισμένων μαθητικών κοινοτήτων, ενώ το 2006 η υπηρεσία έγινε προσβάσιμη σε κάθε άνθρωπο του πλανήτη που η ηλικία του ξεπερνούσε τα 13 χρόνια. Το Facebook είχε το 2013 πάνω από 1 δισεκατομμύριο ενεργούς χρήστες. Επίσης, το Facebook είναι ένα από τα δημοφιλέστερα σιτες για ανέβασμα φωτογραφιών με πάνω από 14 εκατομμύρια φωτογραφίες καθημερινά.[8]

Με αφορμή τη δημοτικότητά του, το Facebook έχει υποστεί κριτική και κατηγορηθεί σε θέματα που αφορούν τα προσωπικά δεδομένα και τις πολιτικές απόψεις των ιδρυτών του. Ωστόσο η συγκεκριμένη ιστοσελίδα παραμένει η πιο διάσημη κοινωνική περιοχή δικτύωσης σε πολλές χώρες ανά το παγκόσμιο. Το Facebook είναι ένας καλός τρόπος δικτύωσης με φίλους και γνωστούς. Παρά το ότι ενέχει κινδύνους (κυρίως για παραβίαση προσωπικών δεδομένων), ο προσεκτικός χρήστης κατά πάσα πιθανότητα δεν θα έχει κάποιο πρόβλημα. Το Facebook ακόμα παρέχει παιχνίδια και υπάρχει η δυνατότητα ανεβάσματος φωτογραφιών και βίντεο.

Τον Απρίλιο του 2011, το Facebook προσέφερε την ευκαιρία σε εμπόρους και εταιρείες να προωθηθούν μέσω αυτού. Η εταιρεία ξεκίνησε την προώθηση της προσκαλώντας μία επιλεγμένη ομάδα Βρετανών διαφημιστών να συναντήσουν τα κορυφαία στελέχη του Facebook σε μία 'σύνοδο κορυφής εμπνευστών' το Φεβρουάριο του 2010. Μέχρι σήμερα το Facebook έχει εμπλακεί σε καμπάνιες για τοTrue Blood, το American Idol και το Top Gear. Μέσα ενημέρωσης όπως η Washington Post, η Finanacial Times και το ABS News έχουν χρησιμοποιήσει συγκεντρωτικά δεδομένα των χρηστών του Facebook ώστε να δημιουργήσουν διάφορες γραφικές παρουσιάσεις δεδομένων και διαγράμματα που συνοδεύουν τα άρθρα τους. [6]

2.1.2 *Twitter*

Το Twitter είναι ένα Κοινωνικό Δίκτυο το οποίο ανήκει στην ευρύτερη κατηγορία των Social Media και θεωρείται το δεύτερο δημοφιλέστερο αυτή τη στιγμή πίσω από το Facebook. Δημιουργήθηκε τον μάρτιο του 2006 από τον Τζακ Ντόρσεϊ και δημοσιεύθηκε τον Ιούλιο του ίδιου χρόνου. Σύμφωνα με τα τελευταία οικονομικά αποτελέσματα, το Twitter έχει μηνιαία 316 εκατομμύρια ενεργούς χρήστες.[8] Αυτό που χαρακτηρίζει το συγκεκριμένο Κοινωνικό Δίκτυο από τα υπόλοιπα, είναι οι δημοσιεύσεις με όριο τους 140 χαρακτήρες, καθώς επίσης και η προώθηση του δημόσιου διαλόγου. Η χρήση του έχει να κάνει κυρίως με την ενημέρωση. Στο Twit-

ter δε συναντάει κανείς φιλίες, αλλά ακόλουθους, ή αλλιώς followers. Δεν είναι απαραίτητο για δύο χρήστες να ακολουθεί ο ένας τον άλλον. Κάθε χρήστης ακολουθεί όποιους θέλει και ακολουθείται από οποιονδήποτε. Αυτό σημαίνει πως ο καθένας διαμορφώνει την αρχική του σελίδα με περιεχόμενο το οποίο επιθυμεί, αν και το Twitter έκανε πρόσφατα κάποιες αλλαγές.

Σε σύγκριση με το Facebook το οποίο φιλτράρει και δείχνει στο χρήστη συγκεκριμένες δημοσιεύσεις στην αρχική σελίδα και όχι όλες, όποιες αυτό θεωρεί σημαντικότερες, στο Twitter εμφανίζονται όλα τα tweets των χρηστών που ακολουθεί κανείς. Μία από τις σημαντικότερες λειτουργίες των Κοινωνικών Δικτύων, τα hashtags, καθιερώθηκαν από το Twitter και χρησιμοποιούνται σε μεγάλο βαθμό για συγκέντρωση όλων των tweets γύρω από μια συζήτηση. Οι κυριότερες λειτουργίες του Twitter είναι το retweet το οποίο χρησιμοποιείται για κοινοποίηση ενός tweet, το favorite το οποίο χρησιμοποιείται περισσότερο ως το like του Facebook, και το reply το οποίο χρησιμοποιείται για απάντηση σε ένα tweet.

Το Twitter είναι ένα μέσο το οποίο ενισχύει το δημόσιο διάλογο, κάτι το οποίο είναι εφικτό αφού δε μπορεί κανείς να φλυαρήσει με το όριο των 140 χαρακτήρων. Τέλος, χρησιμοποιείται από πολλές διάσημες προσωπικότητες και τα περισσότερα έκτακτα νέα και ειδήσεις προέρχονται από αυτό. Στο Twitter είναι όλοι 'δημοσιογράφοι'. [7]

2.1.3 *Google+*

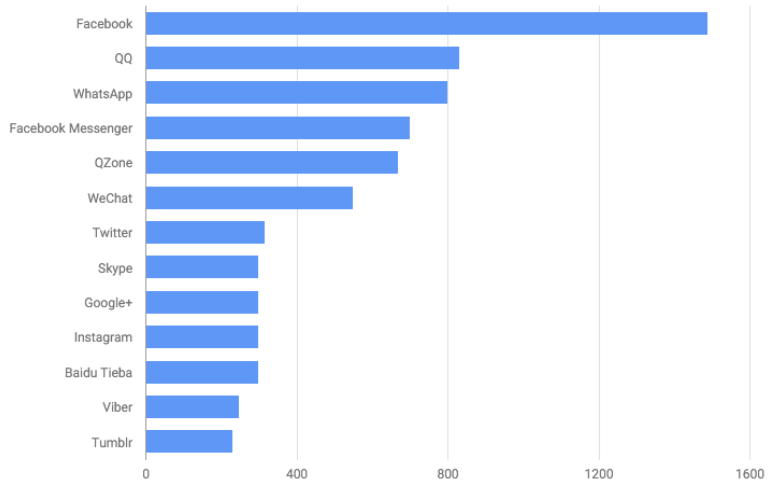
Το Google+ είναι ένα από τα νεότερα κοινωνικά δίκτυα, αφού φέτος έκλεισε τα τέσσερα χρόνια λειτουργίας του. Η Google το θεωρεί περισσότερο ως μία κοινωνική πλατφόρμα πάνω στην οποία έχει συνδέσει όλες τις υπόλοιπες υπηρεσίες της, αφού το προφίλ στο Google+ χρησιμοποιείται για υπηρεσίες όπως το YouTube, Gmail, κλπ. Επίσης, μέσω του συγκεκριμένου κοινωνικού δικτύου έγινε μία προσπάθεια σύνδεσης των σελίδων των διαφόρων sites του ίντερνετ με τους κατόχους τους στην αναζήτηση της Google.

Η τελευταία μέτρηση χρηστών του Οκτώβρη 2013, έδειξε πως το Google+ μετρά πάνω από 540 εκατ. ενεργούς χρήστες μέσω των διαφόρων υπηρεσιών της Google, ενώ αποκλειστικά μέσω του κοινωνικού δικτύου υπάρχουν πάνω από 300 εκατ. ενεργοί χρήστες. [9] Παρόλα αυτά, πολλοί το θεωρούν ως 'πόλη φάντασμα', αφού αρκετοί είναι εκείνοι οι οποίοι κάποια στιγμή δημιούργησαν ένα προφίλ και το παράτησαν γρήγορα, κυρίως επειδή η Google προσπάθησε να επιβάλει τη χρήση του. Αυτό που ξεχωρίζει το Google+ από τα υπόλοιπα κοινωνικά δίκτυα είναι η υπηρεσία Hangouts για βίντεο κλήσεις έως 10 χρηστών και οι αυξημένες δυνατότητες επεξεργασίας φωτογραφιών.

Η Google έχει περιγράψει το Google+ ως ένα 'κοινωνικό στρώμα' που βελτιώνει πολλές από τις διαδυκτιακές υπηρεσίες της, το οποίο δεν είναι

απλά μια ιστοσελίδα κοινωνικής δικτύωσης, αλλά είναι επίσης ένα εργαλείο συγγραφικού δικαιώματος που συνοδεύει περιεχόμενο που βρίσκεται στο Παγκόσμιο Ιστό άμεσα με τον ιδιοκτήτη/συγγραφέα του. Αν και από πολλούς θεωρείται ως ο δεύτερος μεγαλύτερος ιστοχώρος κοινωνικής δικτύωσης στο κόσμο μετά το Facebook, τα περισσότερα site στατιστικών δεδομένων δείχνουν ότι έχει περίπου 300 εκατομμύρια ενεργούς χρήστες το μήνα.[8] Το Google+ περιέχει επίσης σελίδες επιχειρήσεων, όμως το δυνατότερο χαρτί του είναι οι κοινότητες. Θυμίζουν Facebook groups, όμως είναι πιο εύχρηστες αφού μπορούν να περιέχουν πολλές ενότητες θεμάτων και φυσικά η θεματολογία τους ποικίλλει, για την ακρίβεια οποιοδήποτε θέμα μπορείτε να φανταστείτε.

Οι λειτουργίες δημοσιεύσεων τις οποίες χρησιμοποιούν οι χρήστες έχουν να κάνουν με το +1, το οποίο χρησιμοποιείται όπως το like του Facebook, την κοινοποίηση και φυσικά τα σχόλια. Επίσης, διαθέσιμη είναι και η λειτουργία των hashtags. Το Google+ έχει κάνει εξαιρετική δουλειά στον τομέα των φωτογραφιών, αφού μπορεί αυτόματα και τις βελτιώνει μετά το upload του χρήστη, περιέχει εξαιρετικά εργαλεία επεξεργασίας, ενώ επίσης μπορεί και δημιουργεί αυτόματα διάφορα όμορφα εφέ στις φωτογραφίες των χρηστών στις κινητές συσκευές.



Σχήμα 2.1: Μηνιαίοι χρήστες σε κάθε κοινωνικό δίκτυο.

2.2 Μοντελοποίηση κοινωνικών αλληλεπιδράσεων με γράφους

2.2.1 Κύκλοι

Οι κοινωνικοί κύκλοι αποτελούν το σύνολο των δεσμών που αναπτύσσονται μέσω των σχέσεων των ανθρώπων. Είναι ένας ελαστικός όρος με μια ποικιλία ορισμών, σε πολλαπλά επίπεδα. Ο όρος αυτός χρησιμοποιείται από πολλές διαφορετικές κοινωνικές επιστήμες. Η έννοια του όρου αυτού, αναδεικνύει την αξία των κοινωνικών σχέσεων και του ρόλου της συνεργα-

σίας και εμπιστοσύνης. Ένας από τους διαθέσιμους και επικρατέστερους ορισμούς είναι:

'Αποτελεί το σύνολο των πόρων, πραγματικών ή εικονικών, που αποκομίζει ένα άτομο ή μια ομάδα ατόμων, δυνάμει ενός ανθεκτικού δικτύου, με περισσότερο ή λιγότερο θεσμοθετημένες σχέσεις αμοιβαίας γνωριμίας και αναγνώρισης". [39]

Τα πλεονεκτήματα που προκύπτουν από αυτές τις σχέσεις μπορεί να διαφέρουν ως προς την μορφή και τη λειτουργία τους, πάντα με βάση την αλληλοδιασύνδεση μεταξύ αυτών των σχέσεων (σε ατομικό ή συλλογικό επίπεδο). Η έννοια του κοινωνικού κύκλου, για τους χρήστες των μέσων κοινωνικής δικτύωσης επιτρέπει σε αυτούς να αντλήσουν πόρους από τα υπόλοιπα μέλη των κύκλων στα οποία ανήκουν. Οι πόροι αυτοί μπορεί να είναι χρήσιμες πληροφορίες, ανάπτυξη κοινωνικών σχέσεων, καθώς και απόκτηση γνώσης ή και ψυχολογικής υποστήριξης. Θα μπορούσαμε να πούμε ότι ο όρος του κοινωνικού κύκλου χωρίζεται σε δύο κατηγορίες. Η πρώτη κατηγορία αποτελείται από τους κύκλους στους οποίους έχουμε ασθενείς δεσμούς, δηλαδή δεσμούς οι οποίοι αναπτύσσονται ευρέως στην κοινωνία της πληροφόρησης και της ενημέρωσης, αλλά, τα άτομα τα οποία αποτελούν συστατικό μέρος αυτών των δεσμών δεν έχουν κάποια φυσική σχέση.

Στην αντίπερα όχθη έχουμε τους κύκλους, μεταξύ των ατόμων με δυνατές και στενές σχέσεις. Στην κατηγορία αυτή περιλαμβάνονται κυρίως, η έννοια της οικογένειας και οι στενοί φίλοι. Παράλληλα, δημιουργείται και μια τρίτη κατηγορία, η οποία αποτελείται από χρήστες με τους οποίους είχαμε κάποια φυσική σχέση στο παρελθόν. Τα κοινωνικά δίκτυα μας δίνουν την δυνατότητα να είμαστε σε επαφή με αυτούς τους χρήστες, αν και πλέον, δεν έχουμε καμία φυσική σχέση με αυτούς. Τα κοινωνικά δίκτυα προσφέρουν αρκετές δυνατότητες για χρήστες, που σε διαφορετικές περιστάσεις, θα αντιμετώπιζαν δυσκολία στον σχηματισμό και την διατήρηση ισχυρών και ασθενών δεσμών. Το διαδίκτυο μπορεί να βοηθήσει τα όχι τόσο κοινωνικά άτομα, να καλύψουν αυτό το κενό, λόγω των περιορισμένων δεσμών με φίλους και γείτονες. Έτσι, διαπιστώνουμε πως κάποιες μορφές επικοινωνίας, μεσώ υπολογιστή, μπορούν να μειώσουν τα εμπόδια για αλληλεπίδραση και να ενθαρρύνουν περισσότερο τα άτομα με χαμηλή αυτοπεποίθηση και να διευρύνουν τις διαπροσωπικές τους σχέσεις. Εν κατακλείδι, τα μέσα κοινωνικής δικτύωσης, μπορούν να επιτρέψουν συνδέσεις και αλληλεπιδράσεις με χρήστες, που διαφορετικά δεν θα ήταν εύκολο και εφικτό να συμβούν.

Με την πάροδο του χρόνου, καθώς οι σχέσεις διαμορφώνονται ή και καταστρέφονται, οι κοινωνικοί κύκλοι αλλάζουν. Συγκεκριμένα, σημαντικές αλλαγές στη ζωή ενός χρήστη όπως για παράδειγμα να μετακομίσει σε κάποια άλλη πόλη, κατά πάσα πιθανότητα θα δημιουργήσει καινούργιες γνωριμίες με αποτέλεσμα να διαμορφώσει και άλλους κοινωνικούς κύκλους. Συμπεραίνουμε λοιπόν ότι, τα κοινωνικά δίκτυα δεν χρησιμοποιούνται μόνο για τη διατήρηση των υφιστάμενων κοινωνικών κύκλων,

αλλά είναι επίσης εφικτή και η δημιουργία καινούργιων σχέσεων, έτσι ώστε να επεκτείνουν τον κοινωνικό τους κύκλο και σε καινούργιες διαστάσεις.

2.2.2 Γράφοι αλληλεπιδράσεων

Ο στόχος της ανάλυσης της αλληλεπίδρασης των χρηστών του Facebook, Google+ και Twitter είναι να καταλάβουμε πώς οι κοινωνικές συνδέσεις αποτελούν στην πραγματικότητα ενδεικτικό μέτρο για την ενεργό αλληλεπίδραση μεταξύ των χρηστών. Χρησιμοποιώντας δεδομένα από τα κοινωνικά δίκτυα, αποδεικνύεται ότι μερικοί από τους κοινωνικούς δεσμούς αποτελούν ενεργές κοινωνικές σχέσεις. Τα αποτελέσματα αυτά υποδηλώνουν ότι οι κοινωνικοί δεσμοί, και τα κοινωνικά γραφήματα, δεν αποτελούν ακριβείς δείκτες των κοινωνικών σχέσεων μεταξύ των χρηστών.[28] Αυτό έχει σοβαρές επιπτώσεις για την ανερχόμενη γενιά των εφαρμογών που αξιοποιούν τα κοινωνικά γραφήματα.

Για την καλύτερη διαφοροποίηση μεταξύ των ενεργών χρηστών και εκείνων που απλώς συνδέονται, εισάγεται η έννοια των γραφημάτων αλληλεπίδρασης. Ορίζουμε ένα γράφημα αλληλεπιδράσεων ως ένα μη-κατευθυνόμενο γράφημα $G'(n, t) = (V, I)$. Ένα κοινωνικό γράφημα $G = (V, E)$ και γράφημα αλληλεπιδράσεων G' μοιράζονται το ίδιο σύνολο κορυφών V. Εντούτοις, το κοινωνικό γράφημα G χρησιμοποιεί το σύνολο ακμών E (οι οποίες είναι οι κοινωνικές σχέσεις μεταξύ των χρηστών), ενώ το γράφημα αλληλεπιδράσεων G' χρησιμοποιεί το σύνολο ακμών I (το οποίο αποτελεί τις αλληλεπιδράσεις μεταξύ των χρηστών). Το I είναι υποσύνολο του E.[48]

Ένα γράφημα αλληλεπιδράσεων παραμετροποιείται από δύο σταθερές, τις n και t. Αυτές οι σταθερές φιλτράρουν ακμές από το σύνολο των αλληλεπιδράσεων I. Η σταθερά ν ορίζει ένα ελάχιστο αριθμό αλληλεπιδράσεων έτσι ώστε για κάθε ακμή $i_{u,v}$ να ισχύει $|i_{u,v}| \geqslant n$. Για παράδειγμα, αν n=2, τότε η ακμή μεταξύ των χρηστών u και v θα υπάρχει στο γράφημα αλληλεπιδράσεων μόνο αν υπάρχουν 2 ή περισσότερες συνολικές δημοσιεύσεις στο προφίλ των χρηστών u και v. Η παράμετρος t ορίζει ένα χρονικό διάστημα κατά το οποίο πρέπει να έχουν συμβεί οι αλληλεπιδράσεις. Σε συνδυασμό οι παράμετροι n και t θέτουν ένα όριο στο ρυθμό των αλληλεπιδράσεων. Διαισθητικά, ένα γράφημα αλληλεπίδρασης είναι το υποσύνολο του κοινωνικού γραφήματος, η αλληλεπίδραση μεταξύ των τερματικών σημείων κάθε ακμής, είναι μεγαλύτερη ή ίση με το ποσοστό που ορίζεται από το n και το t . Ο βαθμός αλληλεπιδράσεων ενός χρήστη είναι ο αριθμός των κόμβων που γειτνιάζουν με τον χρήστη u στο G' . Αντίστοιχα στο G είναι ο βαθμός ενός κόμβου. [49]

Ορίζουμε τα γραφήματα αλληλεπιδράσεων ως μη κατευθυνόμενα για δύο λόγους. Πρώτον, καθιστώντας τα γραφήματα αλληλεπιδράσεων μη κατευθυνόμενα μας επιτρέπει να τα συγκρίνουμε άμεσα με τα κοινωνικά γραφήματα χρησιμοποιώντας τους ίδιους όρους. Δεύτερον, οι κοινωνικές εφαρμογές έχουν σχεδιαστεί για μη κατευθυνόμενους γράφους, και ως εκ

τούτου πρέπει να ορίσουμε τα γραφήματα αλληλεπιδράσεων με τον ίδιο τρόπο.

Αν και οι αλληλεπιδράσεις είναι εκ φύσεως κατευθυνόμενες, είναι λογικό να τις μοντελοποιούμε ως μη κατευθυνόμενες εάν μπορεί να αποδειχθεί ότι, για ένα συγκεκριμένο σύνολο δεδομένων, οι αλληλεπιδράσεις μεταξύ των χρηστών είναι παρόμοιες. Παρόλα αυτά, κάτι τέτοιο ισχύει περισσότερο στο Facebook και στο Google+ αλλά όχι τόσο στο Twitter. [50]

2.3 Ανάλυση κοινωνικών γράφων και κοινωνικών δικτύων

Η τεχνική ανάλυσης κοινωνικών δικτύων έχει υποστεί μια αναγέννηση με την τεράστια ποσότητα δεδομένων από τα social media και τις ιστοσελίδες. Το περιεχόμενο αυτό είναι μια πλούσια πηγή δεδομένων για την κατασκευή και την ανάλυση των κοινωνικών δικτύων, αλλά είναι τεράστιο σε μέγεθος και είναι από τη φύση του αδόμητο με αποτέλεσμα να παρουσιάζονται πολλαπλές προκλήσεις. Οι τεχνικές αναλύσεις γράφων έχουν αποδειχθεί χρήσιμα εργαλεία για την επίλυση αυτών των προκλήσεων.

Ως συνέπεια της αλλαγής των οικονομικών και της κοινωνικής πραγματικότητας, η αυξημένη διαθεσιμότητα πραγματικών δεδομένων σηματοδοτεί μια νέα εποχή έρευνας και ανάπτυξης των κοινωνικών δικτύων. Η ποσότητα των δεδομένων που δημιουργείται κάθε μέρα από τα social media, ιστοσελίδες και συσκευές κινητής τηλεφωνίας προσφέρει μεγάλες ευκαιρίες και μοναδικές προκλήσεις για την αυτόματη ανάλυση, πρόβλεψη, και περιληπτική παρουσίαση. [13]

Η ανάλυση των κοινωνικών δικτύων είναι η διαδικασία διερεύνησης κοινωνικών δομών μέσω της χρήσης της θεωρίας γραφημάτων και δικτύων. Χαρακτηρίζει δίκτυα τα οποία απεικονίζονται ως γράφοι, έχοντας ως κόμβους εξατομικευμένους παράγοντες, ανθρώπους, ή πράγματα εντός του δικτύου, και ως ακμές τους δεσμούς που τα συνδέουν. Παραδείγματα των κοινωνικών δομών που συνήθως οπτικοποιούνται μέσω της ανάλυσης των κοινωνικών δικτύων περιλαμβάνουν συνήθως κοινωνικά δίκτυα μέσων μαζικής ενημέρωσης, τις σχέσεις φιλίας μεταξύ χρηστών καθώς και οποιεσδήποτε σχέσεις μπορούμε να αναλύσουμε ανάλογα από το κάθε κοινωνικό δίκτυο. Τα δίκτυα είναι συχνά ορατά μέσω διαγραμμάτων sociograms στα οποία οι κόμβοι αναπαρίστανται ως σημεία και οι δεσμοί αναπαρίστανται ως γραμμές.[45, 46, 29]

Έχει αναδειχθεί ως βασική τεχνική στη σύγχρονη κοινωνιολογία. Έχει επίσης αποκτήσει σημαντική θέση στην ανθρωπολογία, βιολογία, μελέτες της επικοινωνίας, την οικονομία, τη γεωγραφία, την ιστορία, την επιστήμη των πληροφοριών, οργανωτικές μελέτες, πολιτικές επιστήμες, κοινωνική ψυχολογία, μελέτες ανάπτυξης, και κοινωνιογλωσσολογία. [34]

Αναπτύχθηκε, αρχικά, σε μη-τεχνική σχετικά μορφή από τα διαρθρωτικά προβλήματα του μεγάλου ανθρωπολόγου Radcliffe-Brown. Από τη δεκαετία του 1930 έως τη δεκαετία του 1970, ένας αυξανόμενος αριθμός κοινω-

νικών ανθρωπολόγων και κοινωνιολόγων άρχισαν να χτίζουν την έννοια της 'κοινωνικής δομής' βασισμένοι στο μοντέλο του Radcliffe-Brown. Αργότερα, κατά τη δεκαετία του 1950, μια μικρή ομάδα ειδικών άρχισαν να ασχολούνται με την εκπόνηση πιο επίσημων τεχνικών ανάλυσης κοινωνικών δικτύων και στις αρχές της δεκαετίας του 1970, εμφανίστηκε μια πληθώρα τεχνικών εργασιών και ειδικών εφαρμογών. Από αυτές τις τεχνικές αναλύσεις, έχουν προκύψει οι βασικές έννοιες της ανάλυσης κοινωνικών δικτύων, και οι τεχνικές έχουν σταδιακά ενσωματωθεί στον βασικό κορμό της ανάλυσης δεδομένων καθώς και σε ένα ευρύτερο πεδίο εφαρμογών.

Παρατηρείται έντονη αύξηση του ενδιαφέροντος για τις τεχνικές ανάλυσης κοινωνικών δικτύων ιδιαίτερα τις δύο τελευταίες δεκαετίες. Το αυξημένο ενδιαφέρον δημιουργήθηκε από την ολοένα και μεγαλύτερη έμφαση στη σημασία της 'δικτύωσης' στην πραγματική ζωή και εν μέρη, από την εξάπλωση των ιστοσελίδων κοινωνικής δικτύωσης, όπως το Facebook, Twitter και Google+. [36]

Ένα τρόπο προσδιορισμού των σχέσεων ανάμεσα σε μια συλλογή αντικειμένων, αποτελούν τα γραφήματα. Ένα γράφημα αποτελείται από ένα σύνολο αντικειμένων, που ονομάζονται κόμβοι, με ορισμένα ζεύγη από αντικείμενα που συνδέονται με δεσμούς και ονομάζονται ακμές. Τα γραφήματα είναι χρήσιμα επειδή λειτουργούν ως μαθηματικά μοντέλα των δομών του δικτύου. Εμφανίζονται σε πολλούς τομείς, όποτε είναι χρήσιμο να προσομοιωθεί το πώς συνδέονται τα διάφορα αντικείμενα μεταξύ τους, είτε φυσικά είτε λογικά. [30] Ένα γράφημα μπορεί να είναι μη κατευθυνόμενο, πράγμα που σημαίνει ότι δεν υπάρχει διάκριση μεταξύ των δύο κορυφών που σχετίζονται με κάθε ακμή, ή κατευθυνόμενο, δηλαδή μια ακμή να κατευθύνεται από μια κορυφή στην άλλη.

Ο βασικός ρόλος της θεωρίας γραφημάτων είναι η απεικόνιση αντικειμένων σε τέτοια μορφή έτσι ώστε να κατανοούνται ευκολότερα. Αφενός, η θεωρητικές εφαρμογές περιλαμβάνουν συνήθως συνδέσεις με τον πραγματικό κόσμο και αφετέρου οι ορισμοί και οι υπολογιστικές μέθοδοι εκφράζονται από τυπικά μαθηματικά. Για πολλούς, αυτή η αλληλεπίδραση είναι που κάνει την θεωρία γραφημάτων τόσο ενδιαφέρουσα. [35]

Ας στραφούμε τώρα σε ορισμένες από τις θεμελιώδεις έννοιες και τους ορισμούς γύρω από τα γραφήματα. Ένα γράφημα το οποίο είναι συνεκτικό και ακυκλικό ονομάζεται δένδρο. Ένα μη συνεκτικό γράφημα χωρίς κύκλους ονομάζεται δάσος. Οι συνεκτικές συνιστώσες ενός δάσους είναι δένδρα. Κορυφή δένδρου με βαθμό ένα ονομάζεται φύλλο. Ένα ιδιαίτερα σημαντικό είδος μη-απλής διαδρομής είναι ένας κύκλος. Ένας κύκλος είναι μία διαδρομή με τρεις τουλάχιστον ακμές, στην οποία ο πρώτος και το τελευταίος κόμβος είναι οι ίδιοι αλλά όλοι οι υπόλοιποι κόμβοι είναι διαφορετικοί.[30]

Δοθέντος ενός γραφήματος, είναι φυσικό να αναρωτηθούμε αν κάθε κόμβος μπορεί να φτάσει σε κάθε άλλο κόμβο από ένα μονοπάτι. Με αυτό κατά νου, μπορούμε να πούμε ότι ένα γράφημα είναι συνδεδεμένο, αν για κάθε ζεύγος κόμβων, υπάρχει ένα μονοπάτι μεταξύ τους. Για παράδειγ-

μα, τα περισσότερα από τα δίκτυα επικοινωνίας και μεταφοράς πρέπει να είναι συνδεδεμένα - ή τουλάχιστον φιλοδοξούμε να είναι συνδεδεμένα - δεδομένου ότι ο στόχος τους είναι να περάσουμε την κυκλοφορία (ή πληροφορία) από τον ένα κόμβο στον άλλο. Από την άλλη πλευρά όμως, δεν υπάρχει λόγος να περιμένουμε ότι τα γραφήματα σε άλλες περιπτώσεις πρέπει να είναι συνδεδεμένα - για παράδειγμα, σε ένα κοινωνικό δίκτυο, θα μπορούσε κάποιος να φανταστεί ότι μπορούν να υπάρχουν δύο άνθρωποι για τους οποίους δεν είναι δυνατόν να κατασκευαστεί μια διαδρομή από ένα κόμβο σε άλλο. [30]

Εάν ένα γράφημα δεν είναι συνδεδεμένο, τότε διασπάται φυσικά σε ένα σύνολο συνδεδεμένων 'κομματιών'. Για να γίνει αυτή η έννοια ακριβής, μπορούμε να πούμε ότι μια συνδεδεμένη συνιστώσα ενός γραφήματος είναι ένα υποσύνολο των κόμβων έτσι ώστε κάθε κόμβος στο υποσύνολο του έχει μια διαδρομή με κάθε άλλο και το υποσύνολο δεν αποτελεί μέρος κάποιου μεγαλύτερου συνόλου με την ιδιότητα ότι κάθε κόμβος συνδέεται με κάποιο άλλο.[30]

Εκτός από το αν δύο κόμβοι συνδέονται με ένα μονοπάτι, είναι επίσης ενδιαφέρων να ρωτήσουμε πόσο μακριά βρίσκετε ο ένας κόμβος από κάποιον άλλο. Στις μεταφορές, τηλεπικοινωνίες ή για το πως εξαπλώνεται μια ασθένεια είναι αρκετά ενδιαφέρων να γνωρίζουμε, για κάτι που ρέει μέσα από το δίκτυο, πόσους κόμβους πρέπει να διασχίσει. [30]

Για να κατανοήσουμε την ανάλυση κοινωνικών δικτύων, πρέπει να κατανοήσουμε τι είναι ένα κοινωνικό δίκτυο, και τι είναι ένας κοινωνικός γράφος. Με απλά λόγια, ένα κοινωνικό δίκτυο είναι απλά ένα δίκτυο οντοτήτων που συνδέονται με κάποιες σχέσεις μεταξύ τους. Φυσικά, οι οντότητες που μας ενδιαφέρουν είναι οι άνθρωποι, και οι σχέσεις που έχουν ιδιαίτερο ενδιαφέρον, κυρίως φιλίες (όπως στο Facebook, Google+, Twitter), τους συναδέλφους (όπως στο LinkedIn), τη συγγένεια, τις επικοινωνίες, και πολλές άλλες κοινωνικές αλληλεπιδράσεις. Στο πλαίσιο της ανάλυσης κοινωνικών δικτύων, ένας κοινωνικός γράφος απεικονίζεται απλά ως ένα διάγραμμα που αντιπροσωπεύει το κοινωνικό δίκτυο. Σε ένα κοινωνικό γράφο, κάθε κόμβος (ή κορυφή) αντιπροσωπεύει ένα πρόσωπο, και μια ακμή μεταξύ δύο κόμβων (άτομα) αντιπροσωπεύει μια σχέση μεταξύ τους. Δεδομένου ότι υπάρχουν πολλές πολύπλοκες σχέσεις μεταξύ των ανθρώπων, υπάρχουν εξίσου πολλοί διαφορετικοί κοινωνικοί γράφοι που αντιπροσωπεύουν αυτές τις σχέσεις.

Ο όρος, κοινωνικό γράφημα, έγινε δημοφιλής στο συνέδριο του Facebook F8 στις 24 μαΐου του 2007, όταν χρησιμοποιήθηκε για να εξηγήσει πώς η νεοεισαχθείσα πλατφόρμα του Facebook θα εκμεταλλευόταν τις σχέσεις μεταξύ των ατόμων, έτσι ώστε να προσφέρει μια πιο πλούσια διαδικτυακή εμπειρία.[44] Από την εισαγωγή της έννοιας του κοινωνικού γραφήματος, από το Mark Zuckerberg, ιδρυτή του Facebook, αποτελεί στόχο του Facebook, να προσφέρει το κοινωνικό γράφημα σε άλλες ιστοσελίδες, έτσι ώστε οι σχέσεις ενός χρήστη να γίνονται γνωστές και να τεθούν

σε χρήση από ιστοσελίδες εκτός του Facebook.[51] Σήμερα, αυτό μπορεί να επιτευχθεί μέσω του Facebook Open Graph API.

Πολλά θέματα έχουν προκύψει όσον αφορά την υπάρχουσα εφαρμογή του κοινωνικού γραφήματος που ανήκει στο Facebook. Για παράδειγμα, επί του παρόντος, η υπηρεσία κοινωνικής δικτύωσης δεν γνωρίζει τις σχέσεις που έχουν αναπτυχθεί μεταξύ των ατόμων σε μια διαφορετική υπηρεσία. Αυτό δημιουργεί μια διαδικτυακή εμπειρία που δεν είναι απρόσκοπτη, λόγω της έλλειψης διαθέσιμων γραφημάτων μεταξύ των υπηρεσιών. Επιπρόσθετα, ένα άλλο πρόβλημα είναι ότι, οι υφιστάμενες υπηρεσίες καθορίζουν τις σχέσεις μεταξύ χρηστών με διαφορετικό τρόπο.

Από το 2010, το κοινωνικό γράφημα του Facebook, είναι το μεγαλύτερο σύνολο δεδομένων κοινωνικού δικτύου στο κόσμο. Περιέχει το μεγαλύτερο αριθμό καθορισμένων σχέσεων μεταξύ του μεγαλύτερου ποσοστού ανθρώπων στη γη, διότι είναι η πιο ευρέως χρησιμοποιούμενη υπηρεσία κοινωνικής δικτύωσης.[22] Πολλοί εξέφρασαν τις ανησυχίες τους για το γεγονός ότι, το κοινωνικό γράφημα του Facebook είναι ιδιοκτησία της εταιρείας και δεν μοιράζεται με άλλες υπηρεσίες, δίνοντας στο Facebook την απόλυτη άδεια να χρησιμοποιεί αυτές τις πληροφορίες ακόμα και όταν κάποιος χρήστης είναι δυσαρεστημένος με το Facebook και αποφασίσει να διαγράψει το λογαριασμό του.

Η Google έχει προσπαθήσει να προσφέρει μια λύση σε αυτό το πρόβλημα με τη δημιουργία του Social Graph API, που κυκλοφόρησε τον Ιανουάριο του 2008.[33] Η συγκεκριμένη υπηρεσία, επιτρέπει στους ιστοχώρους να αντλήσουν δημόσια διαθέσιμες πληροφορίες σχετικά με ένα άτομο για να σχηματίσουν μια ταυτότητα, έτσι ώστε να βελτιωθεί η διαδικτυακή εμπειρία του χρήστη.[23] Ωστόσο, αυτή η προσπάθεια δεν έτυχε της επιδοκιμασίας και της ανταπόκρισης που η Google θα ήθελε με αποτέλεσμα να την αποσύρει το 2012.[24]

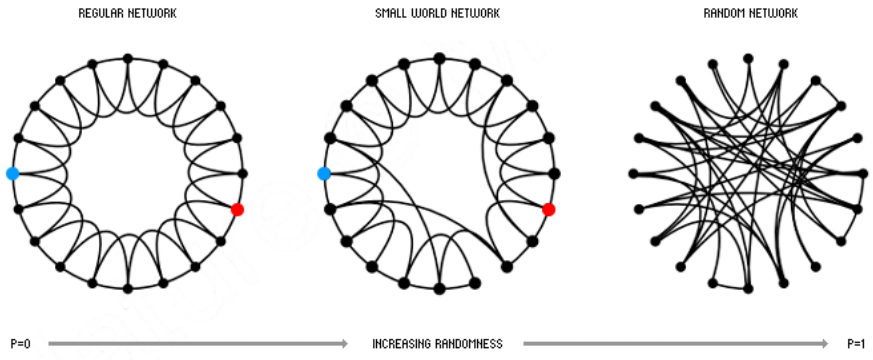
Στα μαθηματικά, ένα γράφημα είναι μια αφαιρετική έννοια για τη μοντελοποίηση σχέσεων μεταξύ οντοτήτων. Δεν είναι διαφορετικό από ένα δίκτυο, το οποίο αποτελεί ένα πιο κοινό όρο για την περιγραφή του ίδιου πράγματος. Όπως αποδεικνύεται, η θεωρία γράφων είναι πολύ ισχυρό εργαλείο για τη μοντελοποίηση φυσικών και τεχνικών συστημάτων. Διαφορετικά πράγματα όπως το διαδίκτυο, δίκτυα παροχής ηλεκτρικής ενέργειας, οικονομίες και ακόμη και τα κύτταρα μπορούν να εκπροσωπούνται και να αναλύονται ως δίκτυα. [12]

Αυτό που είναι επίσης αξιοσημείωτο, είναι ότι πολλά μπορούν να ειπωθούν για ένα γράφημα με την εξέταση και την εξέλιξη της δομής του. Για παράδειγμα, οι επιδημιολόγοι χρησιμοποιούν δομές γράφων για να προβλέψουν την εξάπλωση μιας επιδημίας. Το ίδιο μοντέλο μπορεί να χρησιμοποιηθεί για να καταλάβουμε πώς μια φωτιά εξαπλώνεται. Όσο καλύτερα μπορούμε να κατανοήσουμε τη δομή ενός γράφου που απεικονίζει κάποιο σύστημα, τόσο περισσότερο μπορούμε να το αναλύσουμε, να το προβλέψουμε και να το ελέγξουμε. [12]

Η κοινωνία μας γεννά ένα γιγάντιο κοινωνικό γράφο. Σε αυτό το γράφο, ο καθένας από εμάς είναι ένας κόμβος. Υπάρχει μια σαφής σύνδεση, αν γνωρίζουμε ο ένας τον άλλον. Για παράδειγμα, δύο άτομα μπορούν να συνδέονται επειδή εργάζονται μαζί ή επειδή ήταν συμφοιτητές, είτε επειδή είναι παντρεμένοι. Κάποιες άλλες επιλογές που μας παρέχει το Facebook φαίνονται πιο κάτω.

Σχήμα 2.2: Facebook

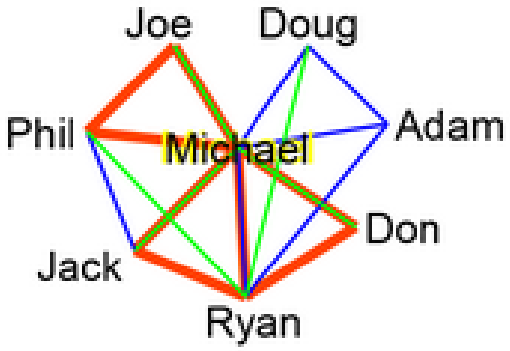
Οι κοινωνιολόγοι έχουν μελετήσει αυτούς τους γράφους για δεκαετίες. Τα κοινωνικά δίκτυα έχουν μια ιδιότητα γνωστή ως 'μικρός Κόσμος' (Small World), που είναι ευρύτερα γνωστή ως 'Οι Έξη βαθμοί διαχωρισμού'. Τόσο ανεπίσημα όσο και ως επιστημονική παρατήρηση, όλοι οι άνθρωποι συνδέονται μεταξύ τους, αλλά όχι περισσότερο από έξι διαδοχικές συνδέσεις. Αυτό συμβαίνει, διότι, με αυτό το τρόπο δημιουργούνται τα ανθρώπινα δίκτυα. Τα ανθρώπινα δίκτυα θεωρούνται πολύ 'πυκνά' και αυτό έχει ως αποτέλεσμα να συνδέονται με συντομεύσεις, για αυτό ισχύει η ιδιότητα του 'μικρού Κόσμου'.



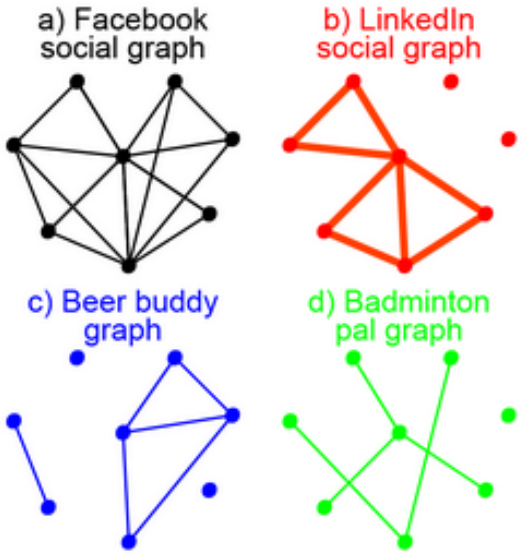
Σχήμα 2.3: Κανονικό δίκτυο, Δίκτυο μικρου Κόσμου, Τυχαίο δίκτυο

Ένας απλός τρόπος για να το παρουσιάσουμε είναι ο εξής, οι φίλοι σας ξέρουν ο ένας τον άλλον, και καθώς περνά ο καιρός, γνωρίζονται μεταξύ τους. Αν τουλάχιστον ένα άτομο από μια ομάδα συναντά κάποιο άλλο άνθρωπο από ένα απομακρυσμένο μέρος του κόσμου, ολόκληρη η ομάδα είναι πλέον συνδεδεμένη με ένα άλλο μέρος του κόσμου.

Ας υποθέσουμε τώρα ότι, ο χρήστης Michael, έχει ένα πολύ μικρό κοινωνικό δίκτυο που αποτελείται από μόνο επτά φίλους του (όπως απεικονίζεται στο πιο κάτω σχήμα). Ας υποθέσουμε ότι έχει μια πολύ απλή ζωή, και έχει μόνο τρεις τύπους κοινωνικών σχέσεων στη ζωή του. Τους συναδέλφους στο χώρο εργασίας (που υποδεικνύεται από τις κόκκινες ακμές), τους φίλους με τους οποίους βγαίνει για ποτό (που υποδεικνύεται από τις μπλε ακμές), και τους φίλους για δραστηριότητες (π.χ μπάντμιντον) (που υποδεικνύεται από τις πράσινες ακμές).



Σχήμα 2.4: Το υποθετικό κοινωνικό δίκτυο του χρήστη Michael



Σχήμα 2.5: Οι διαφορετικοί κοινωνικοί γράφοι του χρήστη Michael

Αν υποθετικά και οι επτά φίλοι του είναι όλοι στο Facebook, τότε ο γράφος, θα μοιάζει με το πιο πάνω σχήμα. Στην περίπτωση αυτή, οι μαύρες

ακμές αντιπροσωπεύουν τη φιλία, ή απλά ανθρώπους που γνωρίζουν ο ένας τον άλλον. Εάν θέλουμε να απεικονήσουμε τους συναδέλφους αυτού του χρήστη, ο γράφος θα μοιάζει όπως το γράφο που απεικονίζεται στη πιο πάνω εικόνα με κόκκινες ακμές. Αντίστοιχα, και για τις υπόλοιπες κατηγορίες φίλων του χρήστη.

Παρατηρήστε ότι έχουμε κατασκευάσει τέσσερις διαφορετικούς κοινωνικούς γράφους από ένα ενιαίο κοινωνικό δίκτυο των ίδιων οκτώ ατόμων. Προσδιορίζοντας ποιες σχέσεις αντιπροσωπεύουν τις ακμές, έχουμε ένα πολύ διαφορετικό γράφο με εντελώς διαφορετικά χαρακτηριστικά. Για παράδειγμα, όταν οι ακμές αντιπροσωπεύουν διασκέδαση δηλαδή όλες οι κατηγορίες φίλων εκτός από τους συναδέλφους, τότε μπορούμε να κατασκευάσουμε ένα ακόμα κοινωνικό γράφο, που θα μοιάζει με μια επικάλυψη των άλλων δύο γράφων. Δεδομένου ότι υπάρχουν πολλές πολύπλοκες σχέσεις μεταξύ των ανθρώπων, πολλοί διαφορετικοί κοινωνικοί γράφοι μπορούν να κατασκευαστούν.

Έτσι, το πιο σημαντικό θέμα όταν αναλύουμε ένα κοινωνικό γράφημα είναι να μάθουμε τι σχέσεις αντιπροσωπεύονται από τις ακμές. Αυτό είναι ακόμη πιο σημαντικό από το τι αντιπροσωπεύουν οι κορυφές, επειδή για την ανάλυση κοινωνικών δικτύων, οι οντότητες που εκπροσωπούνται από τις κορυφές θα είναι συνήθως άτομα.

Για παράδειγμα, ένα σχετικά απλό χαρακτηριστικό των κοινωνικών γράφων είναι η κεντρικότητα κόμβου, η οποία δείχνει πόσες συνδέσεις έχει μια κορυφή. Στο παράδειγμα υπάρχουν επτά μαύρες ακμές στο γράφημα φιλίας που συνδέονται με το κόμβο ‘Michael’, άρα καταλαβαίνουμε ότι έχει επτά φίλους.

Η ερμηνεία των χαρακτηριστικών του γράφου, εξαρτάται επίσης από τη σχέσεις των ακμών. Συνεπώς, δεν μπορούμε να αποφανθούμε για το πόσους συναδέλφους έχει ο συγκεκριμένος χρήστης βασιζόμενοι στο γράφο φιλίας, επειδή η σχέση του συναδέλφου δεν εκπροσωπείται στο γράφο φιλίας. Ως εκ τούτου, δεν μπορούμε να κάνουμε οποιαδήποτε συμπεράσματα για κάποια σχέση ή αλληλεπίδραση για την οποία δεν έχουμε γράφο που οι ακμές εκπροσωπούν αυτή τη σχέση ή αλληλεπίδραση. [11]

2.4 Χαρακτηριστικά γράφων που σχετίζονται με κοινωνικά δίκτυα

Τα δίκτυα έχουν ορισμένα χαρακτηριστικά που μπορεί να χρησιμοποιηθούν για να αναλύσουμε τις ιδιότητες ενός δικτύου και να εξάγουμε κατάλληλα συμπεράσματα. Αυτές οι ιδιότητες του δικτύου συχνά ορίζουν διαφορετικά μοντέλα και μπορεί να χρησιμοποιηθούν για τη σύγκριση διαφορετικών μοντέλων δικτύων. Μερικά από τα χαρακτηριστικά είναι η πυκνότητα, το μέγεθος, βαθμός, μέγεθος μονοπατιού, διάμετρος, συντελεστής ομαδοποίησης, συνεκτικότητα, και κεντρικότητα του δικτύου (ή γράφου)[3]

Ένα δίκτυο (ή γράφημα) είναι ένα (πεπερασμένο) σύνολο κόμβων (σημείων ή κορυφών) και κλάδων (πλευρών ή ακμών) G = V,A, όπου V είναι το

σύνολο των κόμβων και A το σύνολο των κλάδων. Ο κλάδος που συνδέει τους κόμβους i και j συμβολίζεται απλά (i,j). Αν κάθε κλάδος έχει μία συγκεκριμένη διεύθυνση, τότε το δίκτυο ονομάζεται κατευθυνόμενο (directed,oriented), σε αντίθετη περίπτωση ονομάζεται μη κατευθυνόμενο (undirected). Αν κάποιοι κλάδοι έχουν διεύθυνση και κάποιοι όχι, τότε το δίκτυο ονομάζεται μεικτό (mixed). Ένας κλάδος με διεύθυνση (i,j) οδηγεί από τον κόμβο i στον κόμβο j. Δύο κόμβοι που συνδέονται με ένα κλάδο καθώς και δύο κλάδοι που συνδέονται με έναν κόμβο ονομάζονται γειτονικοί. Ο βαθμός ενός κόμβου σε ένα μη προσανατολισμένο δίκτυο είναι ο αριθμός των κλάδων, των οποίων μία κορυφή, είναι αυτός ο κόμβος. Σε ένα προσανατολισμένο δίκτυο, ορίζεται αντίστοιχα ο βαθμός για τον αριθμό των κλάδων που καταλήγουν σε αυτό τον κόμβο και ο βαθμός για τον αριθμό των κλάδων που απομακρύνονται από αυτό τον κόμβο.

Ένα μονοπάτι (ή αλυσίδα) σε ένα μη προσανατολισμένο δίκτυο είναι μία αλληλουχία γειτονικών κλάδων και κόμβων. Σε ένα προσανατολισμένο δίκτυο, τα μονοπάτια έχουν και αυτά διεύθυνση. Ένα μονοπάτι μπορεί να παρασταθεί σαν μία αλληλουχία γειτονικών κόμβων (π.χ. S = a, b, c, . ., i, j, k) ή γειτονικών κλάδων (π.χ. S = (a, b), (b, c), . . ., (i, a), (j, k)]. Ένα μονοπάτι είναι απλό αν κάθε κλάδος εμφανίζεται το πολύ μία φορά στην αλληλουχία και βασικό αν κάθε κόμβος εμφανίζεται το πολύ μία φορά στην αλληλουχία. Κύκλος (ή κύκλωμα) είναι ένα μονοπάτι του οποίου ο αρχικός και ο τελικός κόμβος συμπίπτουν.

Η πυκνότητα δικτύου περιγράφει το μέρος των πιθανών συνδέσεων σε ένα δίκτυο που είναι πραγματικές συνδέσεις. Μια πιθανή σύνδεση είναι μια σύνδεση που θα μπορούσε ενδεχομένως να υπάρχει μεταξύ των δύο κόμβων, ανεξάρτητα από το αν υπάρχει ή όχι στην πραγματικότητα. Ο τύπος για τον υπολογισμό της πυκνότητας είναι ο εξής:

$$PC = \frac{n(n-1)}{2}$$

Όπου το PC είναι οι πιθανές συνδέσεις και το ν είναι ο αριθμός των κόμβων στο δίκτυο. [2]

Το μέγεθος του δικτύου αναφέρεται στον αριθμό των κόμβων N , ή σε κάποιες περιπτώσεις αναφέρεται ως ο αριθμός των ακμών E που μπορεί να κυμαίνονται από N-1 (δέντρο) μέχρι E , δηλαδή πλήρης γράφος. [3] Ο μέσος βαθμός ενός γραφήματος G μας δείχνει πόσες ακμές είναι στο σύνολο E σε σύγκριση με τον αριθμό των κορυφών του συνόλου V.[4] Το μέσο μήκος διαδρομής υπολογίζεται με την εύρεση της συντομότερης διαδρομής μεταξύ όλων των ζευγών των κόμβων, προσθέτοντας τους, και στη συνέχεια διαιρώντας με το συνολικό αριθμό των ζευγών. Αυτό μας δείχνει, κατά μέσο όρο, τον αριθμό των βημάτων που χρειάζεται για να πάμε από ένα κόμβο του δικτύου σε άλλο. Ως ένα άλλο μέσο μέτρησης των διαγραμμάτων δικτύου, μπορούμε να ορίσουμε τη διάμετρο ενός δικτύου ως τη μεγαλύτερη απόσταση όλων των υπολογιζόμενων συντομότερων δρόμων σε ένα δίκτυο. Είναι η συντομότερη απόσταση μεταξύ των δύο πιο απομακρυσμένων κόμβων στο δίκτυο. Με άλλα λόγια, μόλις υπολογιστεί

το συντομότερο μήκος διαδρομής από κάθε κόμβο σε όλους τους άλλους, η διάμετρος είναι η μεγαλύτερη από όλα τα μήκη διαδρομής που έχουν υπολογιστεί. Η διάμετρος αντιπροσωπεύει το γραμμικό μέγεθος ενός δικτύου. [3][5]

Στη θεωρία γραφημάτων, ο συντελεστής ομαδοποίησης αποτελεί το βαθμό στον οποίο οι κόμβοι σε ένα γράφημα τείνουν να συγκεντρώνονται μαζί. Τα στοιχεία δείχνουν ότι στα περισσότερα δίκτυα στο πραγματικό κόσμο, και ειδικότερα στα κοινωνικά δίκτυα, οι κόμβοι τείνουν να ομαδοποιούνται σε μεγάλο βαθμό. Η ομαδοποίηση είναι μια σημαντική ιδιότητα των κοινωνικών δικτύων. Οι άνθρωποι τείνουν να έχουν φίλους που είναι επίσης φίλοι μεταξύ τους, καταλήγοντας σε σύνολα ανθρώπων, μεταξύ των οποίων υπάρχουν πολλές ακμές. Αντίθετα, ένα σύνολο φτιαγμένο από τυχαία επιλεγμένα άτομα θα έχουν πολύ μικρότερο αριθμό ακμών μεταξύ τους, δηλαδή αυτά τα άτομα δεν θα γνωρίζονται. Για τη μέτρηση της ομαδοποίησης σε ένα κοινωνικό (ή άλλου τύπου) δικτύου, ένα κοινό μέτρο σύγκρισης είναι ο συντελεστής ομαδοποίησης.

Ο συντελεστής ομαδοποίησης είναι ένας πραγματικός αριθμός μεταξύ μηδέν και ένα. Μηδέν είναι όταν δεν υπάρχει ομαδοποίηση και ένα όταν έχουμε μέγιστη ομαδοποίηση. Ενώ η ομαδοποίηση σε ένα δίκτυο μπορεί να μετρηθεί με διάφορους τρόπους, ένας κοινός τρόπος είναι να ελέγξουμε για τρίγωνα, δηλαδή να ελέγξουμε ότι όταν δύο ακμές μοιράζονται ένα κόμβο, τότε σε ένα δίκτυο με υψηλό βαθμό ομαδοποίησης, είναι πιθανό ότι ένα τρίτο άκρο υπάρχει τέτοιο ώστε οι τρεις ακμές να σχηματίζουν τρίγωνο. Αν και ο συντελεστής ομαδοποίησης χρησιμοποιείται συχνά, στην πραγματικότητα υπάρχουν δύο παραλλαγές του, που μπορεί να έχουν εντελώς διαφορετικές τιμές και συνεπώς διαφορετικά συμπεράσματα.

Η πρώτη παραλλαγή είναι η εξής. Έστω ότι θέτουμε το συντελεστή ομαδοποίησης $\mathbf{C1}$ ως τη πιθανότητα ότι δύο προσπίπτουσες ακμές συμπληρώνονται από μια τρίτη έτσι ώστε να σχηματίζουν ένα τρίγωνο. Δύο ακμές θεωρούνται προσπίπτουσες όταν συνδέονται στην ίδια κορυφή. Σε αυτή την περίπτωση, μπορούμε να ελέγξουμε εάν μια τρίτη ακμή υπάρχει τέτοια ώστε οι τρεις ακμές να σχηματίζουν ένα τρίγωνο. Η πιθανότητα να υπάρχει μια τρίτη ακμή και να σχηματίζεται τρίγωνο ισούται με το συντελεστή ομαδοποίησης.

Ο καθολικός συντελεστής ομαδοποίησης ορίζεται ως:

$$C = \frac{3 \times \text{αρ. τριγώνων}}{\text{αρ. συνδεδεμένων τριάδων των κορυφών}}$$

Στον πιο πάνω τύπο, μια συνδεδεμένη τριάδα ορίζεται ως ένα συνδεδεμένο υπογράφημα που αποτελείται από τρείς κορυφές και δύο ακμές. Έτσι, κάθε τρίγωνο σχηματίζει τρεις συνδεδεμένες τριάδες, για αυτό προκύπτει ο παράγοντας τρία στον τύπο.

Η δεύτερη παραλλαγή είναι η εξής. Πρώτα, θα ορίσουμε το συντελεστή ομαδοποίησης $\mathbf{C(u)}$ για κάθε κόμβο $\mathbf{u}$ με το ακόλουθο τρόπο. Έστω $\mathbf{C(u)}$ είναι το ποσοστό των γειτόνων του $\mathbf{u}$ που είναι συνδεδεμένοι με αυτόν

και μεταξύ τους. Με άλλα λόγια, C(u) είναι η πιθανότητα ότι δύο φίλοι του u είναι φίλοι μεταξύ τους σε ένα κοινωνικό δίκτυο. Ο τοπικός συντελεστής ομαδοποίησης καθορίζετε ως ο μέσος όρος όλων των τοπικών συντελεστών ομαδοποίησης στο δίκτυο.

Ο τοπικός συντελεστής ομαδοποίησης ορίζεται για κατευθυνόμενους γράφους ως:

$$C_i = \frac{|\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\}|}{k_i(k_i - 1)}$$

Ο τοπικός συντελεστής ομαδοποίησης ορίζεται για μη κατευθυνόμενους γράφους ως:

$$C_i = \frac{2|\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\}|}{k_i(k_i - 1)}$$

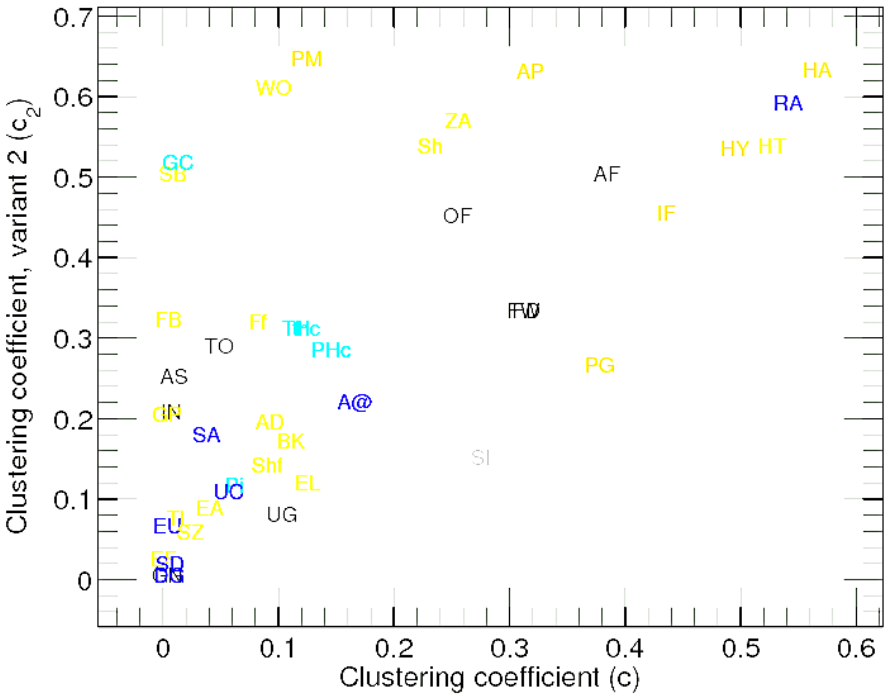
Ο τοπικός συντελεστής ομαδοποίησης μιας κορυφής (κόμβος) σε ένα γράφημα, ποσοτικοποιεί το πόσο κοντά είναι οι γείτονές του έτσι ώστε να αποτελούν ένα πλήρες γράφημα. Ο Duncan J. Watts και ο Steven Strogatz εισήγαγαν το μέτρο το 1998 για να καθοριστεί εάν ένα γράφημα είναι ένα 'μικρό παγκόσμιο δίκτυο'.

Τέλος ο συνολικός μέσος βαθμός ομαδοποίησης σε ένα δίκτυο είναι :

$$C = \frac{1}{n}\sum_{i=1}^{n} C_i$$

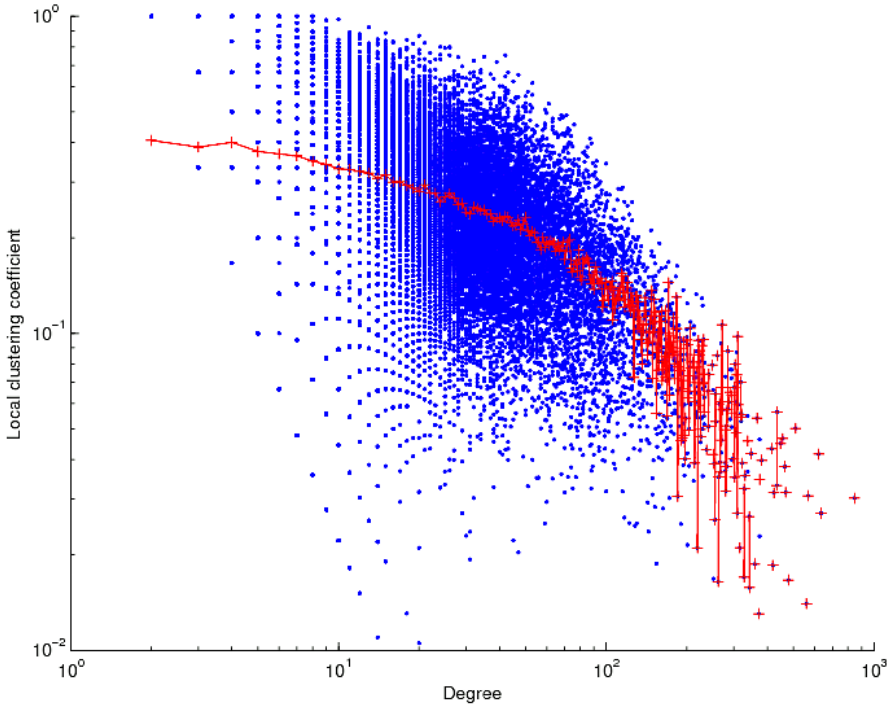
όπου ν είναι ο αριθμός των κόμβων στο δίκτυο.

Και τα δύο μέτρα C1 και C2 υπολογίζουν την ομαδοποίηση σε ένα δίκτυο και ως εκ τούτου θα έπρεπε να συσχετίζονται μεταξύ τους. Όπως, παρατηρούμε όμως στο επόμενο διάγραμμα κάτι τέτοιο δεν ισχύ. Όποτε, θα πρέπει να είμαστε ιδιαίτερα προσεκτικοί για το πια από τις δύο μεθόδους θα χρησιμοποιήσουμε.



Σχήμα 2.6: Καθολικός και Τοπικός βαθμός ομαδοποίησης

Στον άξονα X παρουσιάζουμε τον καθολικό συντελεστή ομαδοποίησης και στον άξονα Υ το τοπικό συντελεστή ομαδοποίησης. Σαφώς, δεν μπορούμε να πούμε ότι οι παραλλαγές συσχετίζονται με οποιονδήποτε τρόπο. Για να δούμε γιατί οι δύο συντελεστές ομαδοποίησης δεν συσχετίζονται, θα πρέπει να σκεφτούμε τον τρόπο που υπολογίζονται. Στην καθολικό συντελεστή ομαδοποίησης, κάθε ζεύγος ακμών που προσπίπτουν έχουν το ίδιο βάρος, ενώ στο τοπικό συντελεστή ομαδοποίησης κάθε κόμβος έχει το ίδιο βάρος. Αυτό σημαίνει ότι οι κόμβοι με υψηλό βαθμό έχουν μεγαλύτερο συντελεστή ομαδοποίησης για τη πρώτη περίπτωση, ενώ μικρότερο στη δεύτερη. Η συσχέτιση των δύο μέτρων μας βοηθά να καταλάβουμε ότι οι κόμβοι με υψηλό και χαμηλό βαθμό σε γενικές γραμμές δεν έχουν τον ίδιο καθολικό και τοπικό συντελεστή ομαδοποίησης.



Σχήμα 2.7: Βαθμός συσχέτισης για ένα δίκτυο από βιβλιογραφικές δημοσιεύσεις

Η πιο πάνω γραφική παράσταση απεικονίζει το βαθμό συσχέτισης για ένα δίκτυο από βιβλιογραφικές δημοσιεύσεις που συνδέονται με κάποια παραπομπή. Δηλαδή, μια σύνδεση από το u στο v σημαίνει ότι η δημοσίευση u παρέθεσε τη δημοσίευση v. Το δίκτυο έχει ληφθεί ως μη κατευθυνόμενο, δηλαδή, αγνοούμε από που παρατίθενται και σε ποιον. Κάθε μπλε κουκίδα στη γραφική παράσταση είναι μία κορυφή του δικτύου. Στον άξονα X έχουμε το βαθμό της κάθε κορυφής και στον άξονα Y έχουμε το τοπικό συντελεστή ομαδοποίησης, και στους δύο άξονες έχουμε λογαριθμικές τιμές. Η κόκκινη γραμμή δείχνει το μέσο τοπικό συντελεστή ομαδοποίησης σε όλους τους κόμβους του ίδιου βαθμού. Αυτό που βλέπουμε εδώ είναι ξεκάθαρο, οι κόμβοι με χαμηλό βαθμό έχουν υψηλό συντελεστή ομαδοποίησης. Με άλλα λόγια, αν ένα άτομο έχει πολλούς φίλους, αυτοί οι φίλοι έχουν λιγότερες ακμές μεταξύ τους, πράγμα που είναι αναμενόμενο δεδομένου ότι ένα άτομο με πολλούς φίλους είναι πιθανό να έχει φίλους από περισσότερες διαφορετικές κοινωνικές ομάδες. [14]

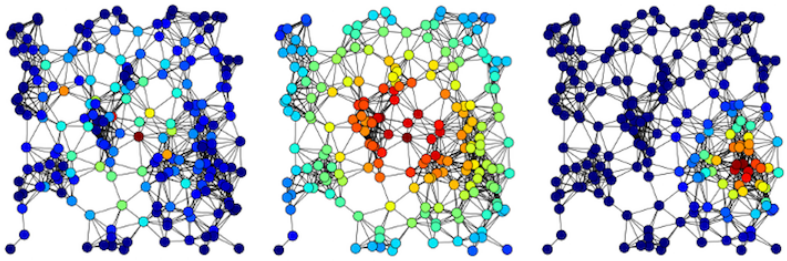
Μία πολύ σημαντική έννοια είναι η συνεκτικότητα του δικτύου. Ένας κόμβος i συνδέεται με τον κόμβο j αν υπάρχει μονοπάτι που να οδηγεί από το σημείο i στο σημείο j. Ένα μη προσανατολισμένο δίκτυο είναι συνεκτικό αν υπάρχει μονοπάτι για κάθε ζεύγος κόμβων του δικτύου.[15] Ένα προσανατολισμένο δίκτυο είναι συνεκτικό αν το αντίστοιχο μη προσανατολισμένο δίκτυο είναι συνεκτικό. Προσέξτε ότι αυτό σημαίνει ότι σε ένα συνεκτικό προσανατολισμένο δίκτυο μπορεί να μην υπάρχει μονοπάτι που να οδηγεί από κάποιο κόμβο i σε κάποιο κόμβο j. Όταν υπάρχει μονοπάτι που να οδηγεί από κάθε κόμβο i σε κάθε άλλο κόμβο j σε ένα προσανατολισμένο δίκτυο, τότε αυτό ονομάζεται ισχυρά συνεκτικό.

Ένα υποδίκτυο G′ = (V′,A′) ενός δικτύου G = (V,A) είναι ένα δίκτυο τέτοιο ώστε $V' \subset V$ και A' ⊂ A. Το σύνολο A' μπορεί να περιέχει μόνο κλάδους μεταξύ σημείων του V'. Ένα δέντρο είναι ένα συνεκτικό γράφημα χωρίς κύκλους. Έτσι, ένα δέντρο σε ένα δίκτυο με n κόμβους περιέχει ακριβώς $n-1$ κλάδους. Επίσης, κάθε ζεύγος κόμβων ενός δέντρου συνδέονται μέσω ενός μοναδικού μονοπατιού. Ένα δέντρο κάλυψης (spanning tree) ενός δικτύου G είναι ένα δέντρο που περιέχει όλους τους κόμβους του G. Σε ένα πρόβλημα δικτύου ορίζονται κάποια χαρακτηριστικά για κάθε κόμβο και κλάδο. Συνήθως υπάρχει κάποιο μέγεθος για κάθε κλάδο του δικτύου το οποίο μπορεί να παριστάνει απόσταση, χρόνο, χωρητικότητα, ροή κτλ. Το μήκος ενός μονοπατιού μεταξύ δύο σημείων του G ισούται προφανώς με το άθροισμα των αποστάσεων όλων των κλάδων στο συγκεκριμένο μονοπάτι. Ο συμβολισμός d(i, j) χρησιμοποιείται συνήθως για να υποδηλώσει την ελάχιστη απόσταση μεταξύ των κόμβων i και j.

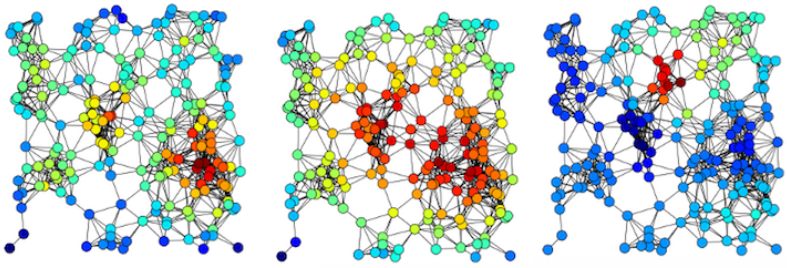
Τέλος, το τελευταίο χαρακτηριστικό που θα δούμε είναι η κεντρικότητα. Η κεντρικότητα, προσπαθεί να ταξινομήσει τους κόμβους ενός δικτύου ανάλογα με το πόσο σημαντικοί θεωρούνται. Υπάρχουν πολλά είδη κεντρικότητας, για παράδειγμα ένας κόμβος ίσως να θεωρείται σημαντικός εάν ενώνει πολλούς άλλους κόμβους, δηλαδή αν ο βαθμός του είναι μεγάλος. Αντιθέτως, μπορούμε να θεωρήσουμε κάποιο κόμβο ότι είναι σημαντικός εάν ενώνει πολλούς άλλους σημαντικούς κόμβους. [32]

Η κεντρικότητα αποτελεί απαραίτητο μέτρο σύγκρισης, όταν η ανάλυση του δικτύου θα πρέπει να απαντήσει σε ερωτήματα όπως, ποιοι κόμβοι του δικτύου θα πρέπει να έχουν ως στόχο να διασφαλίσουν ότι ένα μήνυμα ή πληροφορία εξαπλώνεται σε όλους τους κόμβους του δικτύου. Έχουν καθιερωθεί κάποια συγκεκριμένα χαρακτηριστικά έτσι ώστε να υπάρχει μια ενιαία βάση του τι θεωρούμε σημαντικό κόμβο. [16]

Πιο κάτω παρουσιάζονται τα έξι είδη κεντρικότητας για το ίδιο δίκτυο (γράφημα):



Σχήμα 2.8: A)Betweenness centrality B)Closeness centrality Γ)Eigenvector centrality



Σχήμα 2.9: Δ)Degree centrality E)Harmonic centrality Z)Katz centrality

Μέρος III

ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

3

ΜΗ ΕΠΙΒΛΕΠΟΜΕΝΗ ΜΑΘΗΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ ΜΟΝΤΕΛΟΥ

Η μηχανική μάθηση (machine learning) είναι μια περιοχή της τεχνητής νοημοσύνης η οποία αφορά αλγορίθμους και μεθόδους που επιτρέπουν στους υπολογιστές να 'μαθαίνουν'. Με τη μηχανική μάθηση καθίσταται εφικτή η κατασκευή προσαρμόσιμων (adaptable) προγραμμάτων υπολογιστών, τα οποία λειτουργούν με βάση την αυτοματοποιημένη ανάλυση συνόλων δεδομένων και όχι τη διαίσθηση των μηχανικών που τα προγραμμάτισαν. Η μηχανική μάθηση εφαρμόζεται σε μια σειρά μηχανογραφικών εργασιών όπου η χρήση αλγορίθμων, τόσο κατά το σχεδιασμό όσο και κατά τον προγραμματισμό τους είναι απαραίτητη. Παραδείγματα εφαρμογών αποτελούν το spam filtering, η οπτική αναγνώριση χαρακτήρων, οι μηχανές αναζήτησης και η υπολογιστική όραση. Η μηχανική μάθηση επικαλύπτεται σημαντικά με τη στατιστική, αφού και τα δύο πεδία μελετούν την ανάλυση δεδομένων, όπως επίσης και με τη εξόρυξη δεδομένων (data mining), ωστόσο η δεύτερη εστιάζει περισσότερο στη διερευνητική ανάλυση δεδομένων. Η μηχανική μάθηση και η αναγνώριση προτύπων, μπορούν να θεωρηθούν ως 'οι δυο όψεις του ίδιου τομέα'.[17]

Η μηχανική μάθηση αποτελεί έναν από τους παλαιότερους και σημαντικότερους τομείς έρευνας της Τεχνητής Νοημοσύνης. Στόχος της είναι η δημιουργία συστημάτων που να είναι σε θέση να διδάσκονται από προηγούμενα εμπειρικά δεδομένα, ώστε να εκτελούν την εργασία για την οποία προορίζονται αποτελεσματικότερα. Η διαδικασία εκμάθησης αποτελείται από την απόκτηση εμπειρικών δεδομένων από την αλληλεπίδραση με το περιβάλλον, επεξεργασία των δεδομένων, ούτως ώστε να βρεθούν πιθανές γενικεύσεις ή εξειδικεύσεις και τέλος χρησιμοποίηση των αποτελεσμάτων της επεξεργασίας και λήψη ανατροφοδότησης από το περιβάλλον, έτσι ώστε να βελτιωθεί περαιτέρω το σύστημα. [41]

Έχουν αναπτυχθεί πολλές τεχνικές μηχανικής μάθησης (ταξινόμησης, παλινδρόμησης, ομαδοποίησης, κανόνων συσχέτισης, διαφορικών εξισώσεων) οι οποίες χρησιμοποιούνται ανάλογα με τη φύση του προβλήματος και εμπίπτουν σε ένα από τα παρακάτω δυο είδη:

1. Μάθηση χωρίς επίβλεψη (unsupervised learning) ή μάθηση από παρατήρηση.
2. Επιβλεπόμενη μάθηση (supervised learning) ή μάθηση με παραδείγματα.

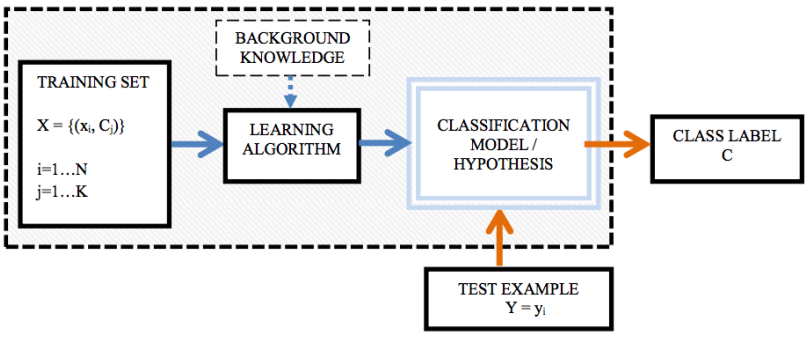
Στη επιβλεπόμενη μάθηση το σύστημα καλείται να 'μάθει' επαγωγικά μέσω ενός συνόλου δεδομένων x, y μια συνάρτηση f, η οποία αποτελεί την περιγραφή ενός μοντέλου. Υπάρχει πάντα κάποιος 'επιβλέπων' ο οποίος πα-

ρέχει τη σωστή τιμή εξόδου y της συνάρτησης για τα δεδομένα που εξετάζονται. Αντίθετα, στη μάθηση χωρίς επίβλεψη το σύστημα δημιουργεί πρότυπα ανακαλύπτοντας συσχετίσεις ή ομάδες σε ένα σύνολο δεδομένων για τα οποία η τιμή εξόδου y της συνάρτησης δεν είναι γνωστή. Το αποτέλεσμα είναι ένα σύνολο προτύπων - περιγραφών, κάθε ένα από τα οποία περιγράφει ένα μέρος των δεδομένων. [26]

3.1 Επιβλεπόμενη μηχανική μάθηση

Μία κατηγορία μηχανικής μάθησης είναι η επιβλεπόμενη μάθηση. Στην επιβλεπόμενη μάθηση υποθέτουμε την παρουσία ενός δασκάλου κατά τη διάρκεια της διαδικασίας εκπαίδευσης. Κάθε δείγμα που χρησιμοποιείται για την εκπαίδευση του συστήματος (δείγμα εκπαίδευσης - **training pattern**) αποτελείται από ένα δείγμα εισόδου (**input pattern**) και ένα δείγμα επιθυμητής εξόδου. Κατά τη διάρκεια της εκπαίδευσης γίνεται σύγκριση μεταξύ της εξόδου που υπολογίζει το σύστημα (δίνοντας του σαν είσοδο το δείγμα εισόδου) και της επιθυμητής εξόδου (όπως αυτή καθορίζεται από το δείγμα επιθυμητής εξόδου) προκειμένου να καθοριστεί το λάθος. Το λάθος στη συνέχεια χρησιμοποιείται για να μεταβληθούν οι ελεύθερες παράμετροι του συστήματος (δηλαδή τα βάρη και τα κατώφλια) έτσι ώστε να βελτιωθεί η απόδοσή του.

Ένα σύστημα που χρησιμοποιεί επιβλεπόμενη μάθηση αρχικά εκπαιδεύεται σε ένα σύνολο παραδειγμάτων εκπαίδευσης τα οποία συνοδεύονται και από τις κατηγορίες στις οποίες ανήκουν. Για παράδειγμα ένα σύστημα παραγωγής ιατρικών διαγνώσεων θα είχε ως παραδείγματα εκπαίδευσης ιατρικές εξετάσεις συνοδευόμενες από τις ορθές διαγνώσεις. Από την εκπαίδευση προκύπτει ένα μοντέλο των κατηγοριών, το οποίο στη συνέχεια χρησιμοποιείται για να κατατάξει νέες περιπτώσεις των οποίων δεν είναι γνωστή η κατηγορία. Υπάρχουν πολλοί γνωστοί αλγόριθμοι επιβλεπόμενης μηχανικής μάθησης, όπως ο αλγόριθμος των k κοντινότερων γειτόνων ($k-NearestNeighbours, k-NN$), ο $NaïveBayes$, ο $ID3$ κ.τ.λ. ([$Mi97$]).



Σχήμα 3.1: Επιβλεπόμενη μάθηση (Supervised Learning)

Οι μέθοδοι που εφαρμόζουν επιβλεπόμενη μάθηση συνήθως χρησιμοποιούνται για ταξινόμηση. Για να ορίσουμε το πρόβλημα της ταξινόμησης ας

υποθέσουμε ότι έχουμε ένα αντικείμενο, το οποίο έχει περιγραφή με διάφορα χαρακτηριστικά (γνωρίσματα, ιδιότητες). Κάθε τέτοιο αντικείμενο μπορεί να ανατεθεί σε μία ακριβώς κατηγορία από ένα πεπερασμένο σύνολο πιθανών κατηγοριών. Τα χαρακτηριστικά είναι ανεξάρτητες παρατηρούμενες μεταβλητές, συνεχείς ή διακριτές. Η κατηγορία είναι μια εξαρτημένη διακριτή μεταβλητή και η τιμή της καθορίζεται από τις τιμές των αντίστοιχων ανεξάρτητων μεταβλητών.

Οι μέθοδοι μηχανικής μάθησης χρησιμοποιούνται μεταξύ άλλων και για την κατασκευή των ταξινομητών. Στόχος των ταξινομητών είναι να καθορίζουν σε ποια κατηγορία πρέπει να ανατεθεί κάθε αντικείμενο.

Στο παράδειγμα που αναφέραμε προηγουμένως, κάθε ασθενής περιγράφεται με συνεχή χαρακτηριστικά (π.χ. ηλικία, ύψος, θερμοκρασία σώματος, καρδιακή συχνότητα, πίεση) και με διακριτά χαρακτηριστικά (π.χ. φύλο, θέση του πόνου). Στόχος του ταξινομητή είναι να παράγει μια διάγνωση (π.χ. υγιής, γρίπη, πνευμονία). Για να καθορίσει την κατηγορία, ένας ταξινομητής χρειάζεται να περιγράψει μια διακριτή συνάρτηση, δηλ. Μια αντιστοίχιση από τον χώρο των χαρακτηριστικών στον χώρο των κατηγοριών.

Αυτή η συνάρτηση μπορεί να δίνεται εκ των προτέρων ή μπορεί να προκύψει από τα παραδείγματα εκπαίδευσης, τα οποία περιγράφουν παλαιότερα λυμένα προβλήματα. Για παράδειγμα, ας θεωρήσουμε την περίπτωση της ιατρικής διάγνωσης. Παλαιότερα λυμένα προβλήματα αποτελούν οι ιατρικές εγγραφές, οι οποίες περιλαμβάνουν τις διαγνώσεις για όλους τους ασθενείς οι οποίοι νοσηλεύτηκαν σε ένα νοσοκομείο. Στόχος του αλγόριθμου μηχανικής μάθησης είναι να καθορίσει την αντιστοίχιση μαθαίνοντας από το σύνολο των ασθενών με γνωστές διαγνώσεις. Αυτή η αντιστοίχιση μπορεί να χρησιμοποιηθεί στη συνέχεια για τη διάγνωση νέων ασθενών.

Στις μεθόδους επιβλεπόμενης μηχανικής μάθησης (ή μάθηση από παραδείγματα) οι αλγόριθμοι λειτουργούν ως εξής. Δεδομένου ενός συνόλου N το οποίο αποτελείται από τα παραδείγματα εκπαίδευσης $\{(x_1, y_2), ..., (x_n y_n)\}$ έτσι ώστε το x_i είναι το χαρακτηριστικό διάνυσμα του ι-οστού παραδείγματος και το y_i είναι ετικέτα, δηλαδή, κατηγορία ένας αλγόριθμος μάθησης αναζητά μια συνάρτηση $g : X \to Y$ όπου το X αποτελεί το σύνολο εισόδων και το Y το σύνολο εξόδων. Η συνάρτηση g είναι ένα στοιχείο του συνόλου των πιθανών συναρτήσεων G, που συνήθως ονομάζεται σύνολο υποθέσεων. Είναι μερικές φορές βολικό να αντιπροσωπεύουμε το g χρησιμοποιώντας μια συνάρτηση βαθμολόγησης $f : X \times Y \to \Re$ έτσι ώστε η συνάρτηση g να ορίζεται ως η συνάρτηση που επιστρέφει την τιμή του Y που δίνει την υψηλότερη βαθμολογία: $g(x) = \underset{y}{\operatorname{argmax}} f(x, y)$. Εστω F χαρακτηρίζει το σύνολο των συναρτήσεων βαθμολόγησης.[18]

Αν G και F μπορεί να είναι σύνολα συναρτήσεων, πολλοί αλγόριθμοι μάθησης αποτελούν πιθανοτικά μοντέλα όπου η g παίρνει τη μορφή ενός δεσμευμένου μοντέλου πιθανοτήτων $g(x) : P(y|x)$ (δηλαδή η πιθανότητα του y δεδομένου του x, ή, η f παίρνει τη μορφή ενός μοντέλου της από κοινού

συνάρτησης πυκνότητας πιθανότητας $f(x, y) = P(x, y)$. Για παράδειγμα, ο Naive Bayes και η linear discriminant analysis είναι μοντέλα της από κοινού συνάρτησης πυκνότητας πιθανότητας, ενώ ο αλγόριθμος logistic regression είναι ένα μοντέλο δεσμευμένης πιθανότητας.

3.2 Μη-Επιβλεπόμενη μηχανική μάθηση

Στην μη-επιβλεπόμενη μάθηση δεν υπάρχει κάποιος δάσκαλος για να εφοδιάσει το σύστημα με τη σωστή απάντηση, δηλαδή τα δείγματα εκπαίδευσης αποτελούνται μόνο από τα δείγματα εισόδου και δεν περιέχουν δείγματα επιθυμητής εξόδου. Στην περίπτωση αυτή, λοιπόν, το σύστημα πρέπει να μάθει ανακαλύπτοντας και προσαρμόζοντας τον εαυτό του σε κάποια δομικά χαρακτηριστικά των διανυσμάτων εισόδου, αυτό γίνεται ανακαλύπτοντας κάποιες στατιστικές κανονικότητες και ομαδοποιήσεις των δειγμάτων εισόδου. Ένα τέτοιο είδος μάθησης επιτυγχάνεται με την ενίσχυση επιλεγμένων βαρών προκειμένου το διάνυσμα εξόδου να ταιριάζει σε κεντρικά πρωτότυπα δείγματα εκπαίδευσης που είναι αντιπροσωπευτικά ενός συνόλου από παρόμοια δείγματα.

Στη μηχανική μάθηση, το πρόβλημα της μάθησης χωρίς επίβλεψη είναι ότι το σύστημα προσπαθεί να βρει κάποια κρυμμένη δομή στα μη ταξινομημένα δεδομένα. Από τη στιγμή που τα παραδείγματα που δίνονται στο σύστημα είναι μη επισημασμένα, δεν υπάρχει σφάλμα ή ανταμοιβή για την αξιολόγηση μιας πιθανής λύσης. Η μάθηση χωρίς επίβλεψη είναι στενά συνδεδεμένη με το πρόβλημα της εκτίμησης πυκνότητας στη στατιστική.[37] Ωστόσο η μάθηση χωρίς επίβλεψη περιλαμβάνει επίσης πολλές άλλες τεχνικές που προσπαθούν να συνοψίσουν και να εξηγήσουν βασικά χαρακτηριστικά των δεδομένων. Πολλές μέθοδοι που χρησιμοποιούνται στην μάθηση χωρίς επίβλεψη βασίζονται σε μεθόδους εξόρυξης δεδομένων (data mining).

Ένα από τα πιο γνωστά είδη μάθησης χωρίς επίβλεψη είναι η ομαδοποίηση (clustering). Σε αυτό το είδος της μάθησης, ο στόχος δεν είναι να μεγιστοποιηθεί η λειτουργία χρησιμότητας, αλλά απλώς να βρεθούν ομοιότητες στα δεδομένα εκπαίδευσης. Για παράδειγμα, ομαδοποιώντας άτομα με βάση τα δημογραφικά στοιχεία θα μπορούσε να οδηγήσει σε μια ομαδοποίηση των πλουσίων στη μία ομάδα και τους φτωχούς στην άλλη. Στόχος του αλγορίθμου μάθησης είναι να καθορίσει συνεκτικά υποσύνολα (συστάδες, clusters) των παραδειγμάτων εκπαίδευσης. Το πλήθος των συστάδων είτε δίνεται εκ των προτέρων ως μέρος της γνώσης που προϋπάρχει είτε καθορίζεται από τον αλγόριθμο μάθησης. Επομένως ο αλγόριθμος πρέπει να καθορίζει ένα σχετικά μικρό πλήθος συνεκτικών συστάδων δηλ. υποσυνόλων παρόμοιων παραδειγμάτων. Η επιλογή του μέτρου ομοιότητας (απόστασης) είναι το πιο σημαντικό κομμάτι της προϋπάρχουσας γνώσης και υψίστης σημασίας για μια επιτυχή και ουσιαστική ομαδοποίηση.

Η μη επιβλεπόμενη μάθηση, χρησιμοποιείται αρκετά και έχει σημειώσει μεγάλη εξέλιξη τα τελευταία χρόνια. Για παράδειγμα η Google χρησιμοποιεί

τεχνικές μη επιβλεπόμενης μάθησης για να αυξήσει τη διάρκεια μπαταρίας στα κινητά που τρέχουν λογισμικό Android M, απενεργοποιώντας κάποιες υπηρεσίες για κάποιο συγκεκριμένο χρονικό διάστημα που γνωρίζει ότι ο χρήστης δεν θα χρησιμοποιήσει το κινητό του. Μπορεί να είναι μια ισχυρή τεχνική όταν υπάρχει ένας εύκολος τρόπος να ανατεθούν τιμές σε γεγονότα. Επίσης, μπορεί να είναι χρήσιμη όταν υπάρχουν αρκετά στοιχεία για να σχηματιστούν οι ομάδες (αν και αυτό αποδεικνύεται ότι είναι δύσκολο μερικές φορές και ειδικά όταν πρόσθετα στοιχεία σχετικά με τα μέλη της συστάδας μπορεί να χρησιμοποιηθούν για την παραγωγή περαιτέρω αποτελεσμάτων.

Οι πιο γνωστές προσεγγίσεις για ομαδοποίηση είναι οι ακόλουθες. Η ιεραρχική ομαδοποίηση (hierarchical clustering) η οποία έχει δύο υποκατηγορίες. Στη συσσωρευτική (bottomup) ιεραρχική ομαδοποίηση, κάθε παράδειγμα αποτελεί αρχικά μια ξεχωριστή συστάδα. Σε κάθε βήμα του αλγόριθμου, οι πιο όμοιες συστάδες συγχωνεύονται, σχηματίζοντας με αυτόν τον τρόπο ένα δέντρο ομαδοποιήσεων (δενδρόγραμμα). Συνήθως η συγχώνευση γίνεται κατά ζεύγη και οδηγεί στο σχηματισμό ενός δυαδικού δέντρου. Η συγχώνευση συνεχίζεται μέχρις ότου όλα τα παραδείγματα να ανήκουν σε μια μοναδική συστάδα. Τελικά ο αλγόριθμος ή ο τελικός χρήστης επιλέγουν το πιο κατάλληλο επίπεδο συσταδοποίησης από το δέντρο που έχει κατασκευαστεί. Στη διαιρετική (top-down) ιεραρχική ομαδοποίηση, όλα τα παραδείγματα ανήκουν αρχικά σε μια συστάδα. Το πλήθος των συστάδων αυξάνεται σε κάθε βήμα του αλγόριθμου διαιρώντας μια υπάρχουσα συστάδα σε (δυο συνήθως) υπό-συστάδες. Η διαδικασία συνεχίζεται μέχρι να σχηματιστεί ένα κατάλληλο πλήθος συστάδων.

Η δεύτερη προσέγγιση είναι η ομαδοποίηση με βάση τη διαμέριση (partitional clustering). Αρχικά, πρέπει να είναι γνωστό το πλήθος c των απαιτούμενων ξένων συστάδων. Δοθέντος ενός κριτηρίου μέτρησης της καταλληλότητας της διαμέρισης των παραδειγμάτων σε c συστάδες, ο αλγόριθμος αναζητά τις βέλτιστες διαμερίσεις των παραδειγμάτων. Η διαδικασία ξεκινά με μια αρχική ομαδοποίηση (η οποία συνήθως σχηματίζεται με τυχαία επιλογή c παραδειγμάτων), και ένα σύνολο μετασχηματισμών για την αλλαγή μίας διαμέρισης σε μια άλλη διαμέριση. Ο αλγόριθμος μάθησης τροποποιεί μια διαμέριση μέχρις ότου κανένας επιτρεπτός μετασχηματισμός να μη μπορεί να βελτιώσει το δοθέν κριτήριο για την καταλληλότητα της διαμέρισης. Μια από τις πιο γνωστές τεχνικές ομαδοποίησης που ανήκει σ' αυτήν την κατηγορία είναι ο αλγόριθμος K-means, οποίος αναζητά K κέντρα τα οποία αντιπροσωπεύουν το σύνολο των αρχικών σημείων.

3.3 Το παραγωγικό μοντέλο και πως εφαρμόζεται στο πρόβλημα μας

Για την κατάταξη των 'φίλων' (ή ακολούθων) του χρήστη σε κατάλληλες ομάδες με χρήση Μη Επιβλεπόμενης Μηχανικής Μάθησης, το υπό εκπαίδευση σύστημα τροφοδοτείται με πραγματικά δεδομένα τα οποία έχουν συλλεγεί από τρία κοινωνικά δίκτυα (Facebook, Twitter, Google+). Ο

αλγόριθμος μη γνωρίζοντας τι αποτελεί σωστή δράση ή επιθυμητή κατάσταση, δε μπορεί να εξάγει τη σωστή συνάρτηση έτσι ώστε να ταξινομήσει τα δεδομένα. Συνεπώς, προσπαθεί να ανακαλύψει ανάμεσα στα δεδομένα που παίρνει σαν είσοδο, κάποιες κρυμμένες δομές που πιθανόν να είναι αυτές που αναζητούμε. Μια από τις διαδεδομένες μεθόδους επίλυσης αυτού του προβλήματος είναι η συσταδοποίηση (clustering) ή με άλλα λόγια Μη Επιβλεπόμενη Μηχανική Μάθηση. Πιο κάτω δίνεται η επεξήγησή των μαθηματικών που κρύβονται πίσω από τον αλγόριθμο μηχανικής μάθησης που θα χρησιμοποιήσουμε.

Στη συνέχεια, περιγράφουμε το μοντέλο των κοινωνικών κύκλων που θα κατασκευάσουμε. Επιθυμούμε ένα μοντέλο κοινωνικών κύκλων με τα εξής χαρακτηριστικά:

1. Κόμβοι μέσα σε κύκλους θα πρέπει να έχουν κοινά χαρακτηριστικά.
2. Διαφορετικοί κύκλοι θα πρέπει να διαμορφώνονται από διαφορετικά χαρακτηριστικά, για παράδειγμα, ένας κύκλος μπορεί να αποτελείται από μέλη μιας οικογένειας και ένας άλλος από τους φοιτητές που φοιτούν στο ίδιο πανεπιστήμιο.
3. Οι κύκλοι επιτρέπεται να αλληλεπικαλύπτονται, και επιτρέπεται επίσης η δημιουργία μικρότερων κύκλων μέσα σε μεγαλύτερους. Για παράδειγμα, ένας κύκλος φίλων από το ίδιο πρόγραμμα σπουδών μπορεί να σχηματιστεί μέσα σε έναν κύκλο από το ίδιο πανεπιστήμιο, όπως φαίνεται πιο πάνω στο Σχήμα 2.
4. Θα θέλαμε να αξιοποιήσουμε τόσο τις πληροφορίες από τα προφίλ των χρηστών, όσο και τη δομή του δικτύου προκειμένου να εντοπίσουμε κύκλους.
5. Στην ιδανική περίπτωση, θα θέλαμε να είμαστε σε θέση να εντοπίσουμε ποια χαρακτηριστικά του προφίλ προκάλεσαν τη δημιουργία ενός κύκλου, έτσι ώστε το μοντέλο είναι ερμηνεύσιμο από τον χρήστη.

Η είσοδος στο μοντέλο μας είναι ένα (κατευθυνόμενο ή μη κατευθυνόμενο) δίκτυο $G = (V, E)$ μαζί με τα στοιχεία από το προφίλ κάθε χρήστη $v \in V$. Ο κεντρικός κόμβος u του δικτύου δεν συμπεριλαμβάνεται στο G. Το δίκτυο G αποτελείται από όλους τους φίλους του χρήστη u. Ορίζουμε με αυτό ακριβώς τον τρόπο το δίκτυο διότι και στη πραγματικότητα κάποιος χρήστης ο οποίος ταξινομεί τους φίλους του σε κύκλους δεν συμπεριλαμβάνει και τον εαυτό του σε αυτούς τους κύκλους. Για κάθε δίκτυο, στόχος μας είναι να προβλέψουμε ένα σύνολο από κύκλους $C = C_1, ..., C_k, C_k \subseteq V$ και να συσχετίσουμε τα παραμετρικά διανύσματα Θ_k τα οποία είναι υπεύθυνα για το πως δημιουργείται κάθε κυκλος.

Κωδικοποιήσαμε τα προφίλ σε ζεύγη χαρακτηριστικών διανυσμάτων $\varphi(x, y)$ έτσι ώστε να ανακαλύψουμε τα κοινά χαρακτηριστικά που ο χρήστης x και ο χρήστης y έχουν. Η παράμετρος Θ_k μας δείχνει ποιες διαστάσεις από αυτά τα $\varphi(x, y)$ διανύσματα είναι σημαντικές για κάθε κύκλο C_k. Αν το χαρακτηριστικό διάνυσμα $\varphi(x, y)$ περιέχει μια παράμετρο η οποία είναι κοινή

και για τους δύο χρήστες (x, y), για παράδειγμα αν οι δύο χρήστες πηγαίνουν στο ίδιο σχολείο και ο κύκλος C_k κατασκευάζεται από χρήστες που φοιτούν σε αυτό το σχολείο τότε η τιμή της παραμέτρου Θ_k πρέπει να είναι ψηλή.

Γενικότερα, πρώτα θα περιγράψουμε ένα μοντέλο το οποίο μπορεί να εφαρμοσθεί χρησιμοποιώντας αυθαίρετα χαρακτηριστικά διανύσματα $\varphi(x, y)$, και αργότερα θα ασχοληθούμε με το πώς να κατασκευάσουμε χαρακτηριστικά διανύσματα από τα προφίλ χρηστών που έχουμε.

Περιγράφουμε ένα μοντέλο κοινωνικών κύκλων το οποίο αντιμετωπίζει τα μέλη του κύκλου ως μη παρατηρούμενες μεταβλητές. Οι κόμβοι σε κάποιο κοινό κύκλο έχουν τη δυνατότητα να δημιουργήσουν ακμές. Αυτό, οδηγεί στο να δημιουργούνται κύκλοι ανάμεσα σε άλλους κύκλους. Ακολούθως, θα κατασκευάσουμε ένα αλγόριθμο μη επιβλεπόμενης μάθησης που θα έχει ως στόχο να βελτιστοποιήσει τις μη παρατηρούμενες μεταβλητές και τα κοινά χαρακτηριστικά των προφίλ έτσι ώστε να ερμηνευθούν καλύτερα τα αποτελέσματα.

Το μοντέλο μας ορίζεται ως εξής. Δίδεται ένα δίκτυο G και ένα σύνολο από Κ κύκλους $C = \{C_1 ... C_k\}$. Μοντελοποιούμε τη πιθανότητα ότι ένα ζεύγος κόμβων $(x, y) \in V \times V$ δημιουργούν μια ακμή ως εξής:

$$p((x,y) \in E) \propto exp\left\{ \underbrace{\sum_{C_k \supseteq \{x,y\}} (\varphi(x,y), \Theta_k)}_{*} - \underbrace{\sum_{C_k \not\supseteq \{x,y\}} a_k(\varphi(x,y), \Theta_k)}_{**} \right\} \tag{3.1}$$

*: Κύκλοι που περιέχουν και τους δύο κόμβους

**: Όλοι οι υπόλοιποι κύκλοι

Για κάθε κύκλο C_k, η παράμετρος Θ_k δείχνει την ομοιότητα μεταξύ προφίλ χρηστών και αποτελεί τη παράμετρο που επιδιώκουμε να μάθουμε. Η κεντρική ιδέα είναι ότι η τιμή του $(\varphi(\xi,\psi),\Theta_k)$ είναι ψηλή αν και οι δύο κόμβοι ανήκουν στον ίδιο κύκλο C_k. Αντίστοιχα, είναι χαμηλή εάν κανένας από αυτούς δεν ανήκει στο κύκλο C_k. Η παράμετρος a_k δείχνει πόσο σημαντικές είναι οι ακμές μέσα και έξω από τους κύκλους που δημιουργούνται. Ως εκ τούτου, η πιθανότητα αυτή δείχνει ότι είναι πιθανότερο να σχηματιστούν ακμές μέσα σε ήδη υπάρχον κύκλους και λιγότερο πιθανόν έξω από αυτούς. Αυτό επιτυγχάνεται με την 'επιβράβευση' ακμών που εμφανίζονται μέσα σε κύκλους σύμφωνα με $\langle \varphi(x,y), \theta_k \rangle$ και 'επιβολής κυρώσεων' των ακμών που εμφανίζονται έξω από τους κύκλους σύμφωνα με $a_k \langle \varphi(x,y), \theta_k \rangle$. Η παράμετρος a_k ρυθμίζει την 'επιβράβευση' ή όχι των ακμών.

Το χαρακτηριστικό διάνυσμα $\varphi(x, y)$ κωδικοποιεί την ομοιότητα μεταξύ προφίλ των δύο χρηστών x και y. Το παραμετρικό διάνυσμα θ_k κωδικοποιεί ποια παρόμοια χαρακτηριστικά ενός προφίλ προκάλεσαν τη δημιουργία κάποιου κύκλου. Ας σημειώσουμε, ότι το ζεύγος (x, y) θα πρέπει

να αντιμετωπίζεται ως μη διατεταγμένο ζεύγος στην περίπτωση ενός μη-κατευθυνόμενου δικτύου (π.χ. Facebook), αλλά θα πρέπει να αντιμετωπίζεται ως ένα διατεταγμένο ζεύγος για τα κατευθυνόμενα δίκτυα (π.χ., το Google+ και το Twitter).

Λαμβάνοντας υπόψη ότι οι ακμές $e = (x, y)$ παράγονται ανεξάρτητα, μπορούμε να γράψουμε την πιθανότητα G όπως φαίνεται πιο κάτω:

$$P_\theta(G;C) = \prod_{e \in E} p(e \in E) \times \prod_{e \notin E} p(e \notin E) \tag{3.2}$$

όπου $\Theta = (\theta_K, a_k)^{k=1...K}$ αποτελεί το σύνολο των παραμέτρων του μοντέλου μας.

Ορίζοντας την συντομογραφία:

$$d_k(e) = \delta(e \in C_k) - \alpha_k \delta(e \notin C_k)$$

$$\Phi(\varepsilon) = \sum_{C_k \in C} d_k(e)(\varphi(e), \vartheta_k)$$

μας επιτρέπει να γράψουμε την πιθανότητα από την εξίσωση (1), ως $p(e \in E) \propto exp\{\Phi(\varepsilon)\}$ ($\delta(c)$ είναι μια συνάρτηση δείκτης που λαμβάνει τιμή 1 όταν η συνθήκη c ικανοποιείται. Σημειώστε ότι ο όρος $\Phi(\varepsilon)$ συμπεριλαμβάνει και το θετικό και το αρνητικό άθροισμα από την εξίσωση (1).

Η πιθανότητα $p(e \in E) \propto exp\{\Phi(\varepsilon)\}$ (τιμές εισόδου απο 0..∞) δεν είναι κανονικοποιημένη. Για να την κανονικοποιήσουμε την θέτουμε ως είσοδο σε μια άλλη συνάρτηση, η οποία αντιστοιχεί τιμές από το 0..1. Αυτό επίσης, καθορίζει εμμέσως την πιθανότητα $p(e \notin E)$. Συγκεκριμένα, έχουμε:

$$p(e \in E) = \frac{e^{\Phi(\varepsilon)}}{1 + e^{\Phi(\varepsilon)}}$$

$$p(e \notin E) = 1 - p(e \in E) = \frac{1}{1 + e^{\Phi(\varepsilon)}}$$

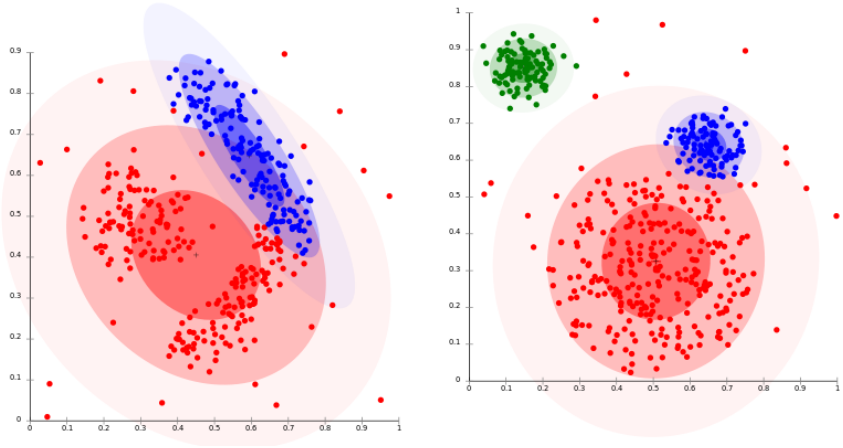
Η λογαριθμική πιθανότητα του G είναι:

$$l_\Theta(G;C) = \sum_{e \in E} \Phi(\varepsilon) - \sum_{e \in V \times V} log(1 + e^{\Phi(\varepsilon)}) \tag{3.3}$$

3.4 Η μέθοδος που χρησιμοποιήσαμε και πως παράγεται το K

3.4.1 *Expectation-maximization*

Ο αλγόριθμος που θα χρησιμοποιήσουμε βασίζεται στο γνωστό αλγόριθμο προσδοκίας-μεγιστοποίησης Expectation-maximization, που κάνει μια αρχική εικασία για τις παραμέτρους, και έπειτα επαναληπτικά βελτιώνει αυτές τις εκτιμήσεις.



Σχήμα 3.2: Expectation-maximizationΠαραδείγματα Συσταδοποίησης

Η γενική ιδέα του αλγορίθμου είναι η εξής:

1. Αντικαθιστά τις ελλείπουσες τιμές με τις κατά εκτίμηση τιμές,
2. Εκτιμά τις παραμέτρους,
3. Επανεκτιμά τις ελλείπουσες τιμές υποθέτοντας ότι οι νέες εκτιμήσεις των παραμέτρων είναι σωστές,
4. Επανεκτιμά τις παραμέτρους και ούτω καθέ εξής, επαναλαμβάνοντας την προαναφερθείσα διαδικασία μέχρι να πετύχουμε σύγκλιση.

Κάθε επανάληψη του αλγορίθμου EM αποτελείται από δύο βήματα: ένα βήμα E (Προσδοκία) που ακολουθείται από ένα βήμα M (Μεγιστοποίηση). Στο βήμα E, η αναμενόμενη τιμή του λογαρίθμου πιθανοφάνειας του πλήρους συνόλου δεδομένων προκύπτει, λαμβάνοντας υπόψη τα παρατηρηθέντα στοιχεία και τις κατ΄ εκτίμηση παραμέτρους από μια προηγούμενη επανάληψη. Στο βήμα M, η δεσμευμένη αναμενόμενη τιμή του λογαρίθμου πιθανοφάνειας του πλήρους συνόλου δεδομένων μεγιστοποιείται. Η τιμή αυτή αυξάνεται έως ότου επιτυγχάνεται ένα στάσιμο σημείο. Με άλλα λόγια, ο αλγόριθμος συνεχίζεται έως ότου η παρατηρηθείσα πιθανοφάνεια που παράγεται σε δύο διαδοχικές επαναλήψεις είναι σχεδόν ίδια.

Συνεπώς, αντιμετωπίζοντας τους κύκλους C ως μη παρατηρούμενες μεταβλητές, προσπαθούμε να βρούμε το $\hat{\Theta} = \{\hat{\Theta}, \hat{a}\}$ έτσι ώστε να μεγιστοποιήσουμε τη κανονικοποιημένη λογαριθμική πιθανότητα της εξίσωσης 3:

$$\hat{\Theta}, \hat{C} = \underset{\Theta,}{argmax} l_{\Theta}(\Gamma \cdot) - \lambda\Omega(\vartheta) \tag{3.4}$$

Επιλύσαμε αυτό το πρόβλημα αυξάνοντας ταυτόχρονα το Θ και C.[40] Δηλαδή, προσπαθήσαμε να ταιριάξουμε τη λανθάνουσα μεταβλητή (τα μέλη του κύκλου) και τη παράμετρο του μοντέλου μας (Θ). Στην πράξη, τα ακόλουθα δύο βήματα επαναλαμβάνονται μέχρι να υπάρξει η επιθυμητή σύγκλιση:

$$C^t = \underset{C}{argmax} l_{\theta^t}(G; C) \tag{3.5}$$

$$\Theta^{t+1} = \underset{\theta}{argmax} l_\theta(G; C^t) - \lambda\Omega(\vartheta) \tag{3.6}$$

Βελτιστοποιήσαμε την εξίσωση 6 με τη χρήση Limited-memory BFGS (αποτελεί αλγόριθμο βελτιστοποίησης που ανήκει στην οικογένεια των μεθόδων quasi-Newton και προσεγγίζει τον αλγόριθμο Broyden-Fletcher-Goldfarb-Shanno (BFGS)) χρησιμοποιώντας περιορισμένο χώρο μνήμης στον υπολογιστή. Αποτελεί ένα δημοφιλή αλγόριθμο για την εκτίμηση παραμέτρων στη μηχανική μάθηση. Υπολογίζοντας τις μερικές παραγώγους της εξίσωσης 6 έχουμε:

$$\frac{\partial l}{\partial \Theta_k} = \sum_{e \in V \times V} - d_e(k)\varphi(e)_k \frac{e^{\Phi(\varepsilon)}}{1 + e^{\Phi(\varepsilon)}} + \sum_{e \in E} d_k(e)\varphi(e)_k - \frac{\partial \Omega}{\partial \Theta_k} \tag{3.7}$$

$$\frac{\partial l}{\partial \alpha_k} = \sum_{e \in V \times V} \delta_e(e \notin C_k)\langle \varphi(e), \Theta_k \rangle \frac{e^{\Phi(\varepsilon)}}{1 + e^{\Phi(\varepsilon)}} + \sum_{e \in E} \delta(e \notin C_k)\langle \varphi(e), \Theta_k \rangle \tag{3.8}$$

Για να βελτιστοποιήσουμε την εξίσωση 6, παρατηρούμε ότι για σταθερό λόγο C/C_i, μπορούμε να λύσουμε την $argmax_{C_i} l_\Theta(G; C/C)$ εκφράζοντας την ως ψευδό-δυαδική βελτιστοποίηση. Η ψευδό-δυαδική βελτιστοποίηση αναφέρεται σε προβλήματα που ορίζονται σε δυαδικές μεταβλητές. Χρησιμοποιήσαμε δυαδικές μεταβλητές, για να καθορίσουμε κατά πόσον ή όχι ένας κόμβος είναι μέλος ενός συγκεκριμένου κύκλου.

Πιο συγκεκριμένα, θα δείξουμε στη συνέχεια ότι σε αυτό το πλαίσιο, βελτιστοποιώντας τα μέλη ενός συγκεκριμένου κύκλου μπορεί να γραφτεί στη μορφή

$$\underset{X}{argmax} \sum_{x_i, x_j \in X} E_{i,j}(x_i, x_j)$$

Όπου x_i και x_j είναι δυαδικές μεταβλητές, και $E_{i,j} : \{0,1\}^2 \to \mathbb{R}$ είναι η ενέργεια ανά ζεύγη. Αυτή είναι η γενική μορφή της ψευδό-δυαδικής βελτιστοποίησης, όπου βελτιστοποιούμε μια πραγματική τιμή σε δυαδικές μεταβλητές. Στη συνέχεια, δείχνουμε ότι η βελτιστοποίηση των μελών ενός κύκλου μπορεί να εκφραστεί με τη μορφή αυτή.

Οι δυαδικές μεταβλητές, κωδικοποιούν τα μέλη των κόμβων σε κάποιο συγκεκριμένο κύκλο C_k. Υπάρχουν τέσσερις πιθανές περιπτώσεις που πρέπει να εξετάσουμε,

1. Μια ακμή εμφανίζεται έξω από τον κύκλο,
2. Μια ακμή που δεν ανήκει στο G εμφανίζεται έξω από τον κύκλο,
3. Μια ακμή εμφανίζεται μέσα στον κύκλο,
4. Μια ακμή που δεν ανήκει στο G εμφανίζεται μέσα στον κύκλο.

Η περίπτωση 'έξω από το κύκλο' συμπεριλαμβάνει και τη περίπτωση όπου ο ένας κόμβος είναι μέλος του κύκλου ενώ ο άλλος όχι. Διαισθητικά, θέλουμε να μεγιστοποιήσουμε την ενέργεια και, επομένως, οι περιπτώσεις

(2) και (3) θα πρέπει να έχουν υψηλή ενέργεια (προτιμούμε ακμές μέσα σε κύκλους), ενώ οι περιπτώσεις (1) και (4) θα πρέπει να έχουν χαμηλή ενέργεια (δεν θέλουμε πολλές ακμές έξω από τους κύκλους). Η ενέργεια θα πρέπει επίσης να εξαρτάται από το $\langle \varphi(\varepsilon), \theta_k \rangle$, το οποίο κωδικοποιεί πόσο συμβατά είναι τα χαρακτηριστικά των i, j με τις παραμέτρους θ_k του κύκλου. Για να επιτευχθεί αυτό, ορίσαμε πρώτα (για ένα ζεύγος κόμβων e):

$$o_k(e) = \sum_{C_i \in C/C_k} d_k(e) \langle \varphi(\varepsilon), \theta_k \rangle$$

Όπου προσθέτουμε όλους τους κύκλους εχτός τον κύκλο C_k του οποίου προσπαθούμε να βελτιστοποιήσουμε τα μέλη του. Αυτή η έκφραση, θα πάρει θετικές τιμές όταν οι κόμβοι στα τελικά σημεία μιας ακμής $e = (i, j)$ ανήκουν σε πολλούς κοινούς κύκλους (εχτός από τον C_k), διαφορετικά θα πάρει αρνητικές.

Ακολούθως, ορίζουμε την ενέργεια ανά ζεύγη E_e^k (παραμετροποιημένη στο κύκλο k, και στο ζεύγος των κόμβων $e = (x, y)$) ως εξής:

$$E_e^k(0,0) = E_e^k(0,1) = E_e^k(1,0) = \begin{cases} o_k(e) - a_k \langle \varphi(e), \theta_k \rangle - \log(1 + e^{o_k(e) - a_k \langle \varphi(e), \theta_k \rangle}), & e \in E \\ -log(1 + e^{o_k(e) - a_k \langle \varphi(e), \theta_k \rangle}), & e \notin E \end{cases}$$

$$E_e^k(1,1) = \begin{cases} o_k(e) + \langle \varphi(e), \theta_k \rangle - \log(1 + e^{o_k(e) - a_k \langle \varphi(e), \theta_k \rangle}), & e \in E \\ -log(1 + e^{o_k(e) + \langle \varphi(e), \theta_k \rangle}), & e \notin E \end{cases}$$

$E_e^k(1,1)$ αντιστοιχεί στην περίπτωση όπου και οι δύο κόμβοι ανήκουν στον ίδιο κύκλο, ενώ στις άλλες τρεις περιπτώσεις, τουλάχιστον ένας κόμβος δεν ανήκει στον κύκλο. Σημειώστε ότι αυτό περιγράφει τις τέσσερις πιθανές περιπτώσεις όπως περιγράφηκε προηγουμένως (περιπτώσεις 1 έως 4 είναι οι τέσσερις εκφράσεις στην προηγούμενη εξίσωση, αντίστοιχα). Τέλος, το πρόβλημα βελτιστοποίησης για να συμπεράνουμε τα μέλη ενός κύκλου γίνεται:

$$C_k = \underset{C}{argmax} \sum_{(x,y) \in V \times V} E_{(x,y)}^k (\delta(x \in C), \delta(y \in C)) \tag{3.9}$$

Εκφράζοντας το πρόβλημα με αυτή τη μορφή, μπορούμε να αξιοποιήσουμε τις υπάρχουσες υλοποιήσεις για την ψευδό-δυαδική βελτιστοποίηση. Τα προβλήματα βελτιστοποίησης αυτής της μορφής είναι γνωστό ότι είναι NP-hard σε γενικές γραμμές. Παρόλα αυτά, υπάρχουν αρκετοί αποδοτικοί αλγόριθμοι προσέγγισης. Χρησιμοποιήσαμε τον αλγόριθμο βελτιστοποίησης ανοικτού κώδικα QBPO (Quadratic Pseudo-Boolean Optimization), ο οποίος είναι είναι σε θέση να προσέγγιση με ακρίβεια προβλήματα της μορφής όπως η εξίσωση (9). Ουσιαστικά, τα προβλήματα του τύπου που παρουσιάζεται στην εξίσωση (9) μειώνονται σε προβλήματα μέγιστης ροής. Τέτοιοι αλγόριθμοι έχουν πολυπλοκότητα $O(|N|^3)$, αν και ο χρόνος εκτέλεσης μέσης περίπτωσης είναι αρκετά καλύτερος. Λύνουμε την εξίσωση (9) για κάθε κύκλο C_k με τυχαία σειρά. Τα δύο βήματα βελτιστοποίησης

των εξισώσεων (5) και (6) επαναλαμβάνονται μέχρι να συγκλίνουν, δηλαδή όταν $C^{t+1} = C^t$. Τέλος, η εξίσωση (4) γίνεται,

$$\Omega(\vartheta) = \sum_{k=1}^{K} \sum_{i=1}^{|\theta_k|} |\vartheta_{ki}|$$

η οποία οδηγεί σε αραιές (και εύκολα ερμηνεύσιμες) παραμέτρους. Ο αλγόριθμος μπορεί να χειριστεί εύκολα σχεδόν όλα τα δίκτυα εκτός από αυτά που είναι αρκετά μεγάλα σε μέγεθος. Στην περίπτωση του Facebook, ο μέσος όρος ενός δικτύου είναι περίπου 190 κόμβοι, ενώ το μεγαλύτερο δίκτυο που συναντήσαμε έχει 4.964 κόμβους. Δεδομένου ότι η μέθοδος αυτή είναι χωρίς επίβλεψη, εξετάζουμε κάθε δίκτυο ξεχωριστά. Αυτό σημαίνει ότι η μέθοδος μας θα μπορούσε να εκτελεστεί σε ολόκληρο το δίκτυο του Facebook, διότι οι κύκλοι ανιχνεύονται ανεξάρτητα για κάθε χρήστη, και τα δίκτυα τυπικά περιέχουν μόνο μερικές εκατοντάδες κόμβους. Πιο κάτω, παρουσιάζουμε τον ψευδοκώδικα του αλγόριθμου.

ALGORITHM 1: Predict complete circles with hyperparameters λ, K.

Data: ego-network $G = (V, E)$, edge features $\phi(e) : E \to \mathbb{R}^F$, hyperparameters λ, K
Result: parameters $\hat{\Theta} := \{(\hat{\theta}_k, \hat{\alpha}_k)\}^{k=1\dots K}$, communities $\hat{C}$
initialize $\theta_k^0 \in \{0,1\}^F$, $\alpha_k^0 := 1$, $C_k := \varnothing$, $t := 0$;
repeat
  for $k \in \{1 \dots K\}$ **do**
    $C_k^t := \text{argmax}_C \sum_{(x,y)\in V\times V} E^k_{(x,y)}(\delta(x \in C), \delta(y \in C))$;
    `// using QPBO, see (eq. 9)`
  end
  $\Theta^{t+1} := \text{argmax}_\Theta l_\Theta(G; C^t) - \lambda\Omega(\theta)$;
  `// using L-BFGS, see (eqs. 7 and 8)`
  $t := t + 1$;
until $C^{t+1} = C^t$;

Σχήμα 3.3: Πρόβλεψη κύκλων με υπερπαραμέτρους λ, K

Το κριτήριο BIC έχει εφαρμοστεί με επιτυχία σε αρκετούς αλγόριθμους ομαδοποίησης για τον καθορισμό του βέλτιστου αριθμού ομάδων. Για να επιλέξουμε αυτόματα το βέλτιστο αριθμό κύκλων, επιλέγουμε κάποιο K έτσι ώστε να ελαχιστοποιεί το εσωτερικό κριτήριο εγκυρότητας Bayesian Information Criterion (BIC). [27] Συνεπώς, το εσωτερικό κριτήριο εγκυρότητας BIC ορίζεται ως:

$$BIC(K; \Theta^K) \simeq -2l_{\Theta^K}(G; C) + |\Theta^K| log|E| \tag{3.10}$$

όπου Θ^K είναι το σύνολο των παραμέτρων που προβλέψαμε όταν υπάρχουν K κύκλοι, και $|\Theta^K|$ είναι ο αριθμός των παραμέτρων. Στη συνέχεια, επιλέγουμε K, έτσι ώστε να ελαχιστοποιηθεί η συνάρτηση:

$$\hat{K} = \underset{K}{argmin} BIC(K; \Theta^K) \tag{3.11}$$

με άλλα λόγια, ένας επιπλέον κύκλος θα προστεθεί στο μοντέλο, μόνο αν κάτι τέτοιο έχει σημαντικές επιπτώσεις στην λογαριθμική πιθανοφάνεια. Η παράμετρος κανονικοποίησης ορίζεται ως $\lambda \in \{0, 1, 10, 100\}$.

Συνεπώς, σε ένα πιο αφαιρετικό επίπεδο τα μαθηματικά που κρύβονται πίσω από τον αλγόριθμο που χρησιμοποιήσαμε περιγράφονται πιο κάτω.

Έστω $G = (V, E)$ αποτελέι γράφο όπου $E \subseteq V \times V$είναι το σύνολο των ακμών. Επίσης μπορεί ισοδύναμα να παρουσιαστεί ως ένας τετραγωνικός πίνακας Α $|V| \times |V|$, όπου

$$A(x,y) = \begin{cases} 1 & (x,y) \in E \\ 0 & (x,y) \notin E \end{cases}$$

Άς υποθέσουμε επίσης ότι, $S_k \subseteq V$είναι οι Κ κύκλοι με την αντίστοιχη συνάρτηση των μελών $\xi^\kappa(x)$ (που αποτελούν το κύκλο) όπου $x \in V$ λαμβάνει τη τιμή 1 μόνο όταν το $x \in S_k$. Υποθέτουμε ότι το μεντέλο για τη παραγωγή των ακμών είναι της μορφής

$$P((x,y) \in E) = P_1(x,y) = \frac{exp(\Phi(\xi,\psi))}{1 + exp(\Phi(\xi,\psi))}$$

$$P((x,y) \notin E) = P_0(x,y) = \frac{1}{1 + exp(\Phi(\xi,\psi))}$$

Η πιθανότητα για το γράφο (που θα ελαχιστοποιηθεί) είναι $P(G) = \prod_{(x,y)\in E} P_1(x,y) \prod_{(x,y)\notin E} P_0(x,y)$ με την αντίστοιχη λογαριθμική πιθανότητα να είναι $-log(P(G)) = -\sum_{x,y} A(x,y)\Phi(x,y) + log(1 + exp(\Phi(x,y)))$. Για τη συνάρτηση $\Phi(x,y)$ επιλέξαμε μια απλή αναπαράσταση η οποία είναι το άθροισμα των δύο πιο κάτω όρων. Ο πρώτος όρος επεξηγεί τη παραγωγή ακμής από την ύπαρξη κύκλων και είναι:

$$\sum_k \xi^k(x)\xi^k(y)\langle\varphi(x,y), a_k\rangle$$

Ο δεύτερος όρος επεξηγεί τη παραγωγή ακμής από τη μη ύπαρξη κύκλων και είναι:

$$\sum_k (1 - \xi^k(x)\xi^k(y))\langle\varphi(x,y), b_k\rangle$$

Προσθέτωντας τον όρο LASSO (βλέπε υποκεφάλαιο 3.5.2) $\Omega(a,b) = \sum_k (||a_k||_1 + ||b_k||_1)$ πετυχαίνουμε κανονικοποίηση. Συνεπώς έχουμε τη συνάρτηση κόστους $J(a,b,\xi) = -log(P(G)) + \lambda\Omega(a,b)$ ή οποία μπορεί να επιλυθεί εφαρμόζοντας τη χρήση ενός συνδυασμού επαναλαμβάνόμενης βελτιστοποίησης για τη προσεγγιστική μέθοδο των κύκλων όπως περιγράψαμε προηγουμένως.

Για το γενικό πρόβλημα έχουμε:

$$J(x) = f(x) + \lambda||\chi||_1$$

επιτυγχάνουμε λύση με επαναλαμβανόμενη εφαρμογή του συντελεστή

$$Prox_\mu(x)_j = max(0, (1 - \frac{\mu}{|x_j|})x_j)$$

Συγκεκριμένα έχουμε

$$x(n+1) = Prox_{\frac{\lambda}{L}}(x(n) - \frac{1}{L}\nabla f(x(n))$$

όπου $L = max_x||\nabla f(x)||_{l^2}$. Τέλος έχουμε

$$\Theta(\xi,\psi) = \frac{exp(\Phi(x,y)}{1+exp(\Phi(\xi,\psi»}$$

$$\frac{\partial J}{\partial a_k} = \sum_{x,y}(\Theta(x,y) - A(x,y))\xi^k(x)\xi^k(y)\phi(x,y)$$

$$\frac{\partial J}{\partial b_k} = \sum_{x,y}(\Theta(x,y) - A(x,y))(1-\xi^k(x)\xi^k(y))\phi(x,y)$$

Συνεπώς έχουμε:

$$\|\frac{\partial J}{\partial a_k}\|_{\ell^2} \leq |V|^2 \max_{x,y} \|\phi(x,y)\|_{\ell^2}$$

$$\|\frac{\partial J}{\partial b_k}\|_{\ell^2} \leq |V|^2 \max_{x,y} \|\phi(x,y)\|_{\ell^2}$$

και μπορούμε να επιλέξουμε:

$$L = \sqrt{(2)}|V|^2 \max_{x,y} \|\phi(x,y)\|_{\ell^2}$$

3.5 Παλινδρόμηση

Η παλινδρόμηση είναι μία στατιστική τεχνική για την εκτίμηση των σχέσεων μεταξύ των μεταβλητών μιας συνάρτησης. Συνήθως η παλινδρόμηση δείχνει πως μεταβάλλεται η τιμή μιας εξαρτημένης μεταβλητής όταν αλλάζει η τιμή μιας ανεξάρτητης μεταβλητής, ενώ η τιμή των υπόλοιπων ανεξάρτητων μεταβλητών παραμένει σταθερή. Ο στόχος της παλινδρόμησης είναι η εύρεση μιας προσεγγιστικής συνάρτησης που περιλαμβάνει όλες τις ανεξάρτητες μεταβλητές. Κάποιες από τις πιο γνωστές υλοποιήσεις της είναι η μέθοδος ελαχίστων τετραγώνων και η γραμμική παλινδρόμηση.

Η τεχνική της παλινδρόμησης χρησιμοποιείται ευρέως για εφαρμογές προβλέψεων και φυσικά στη μηχανική μάθηση. Η συγκεκριμένη τεχνική αποτελεί είναι ένα πολύ ισχυρό εργαλείο, αλλά η αποτελεσματικότητα της εξαρτάται από τον τρόπο που θα επεξεργαστούμε τα δεδομένα αλλά και κατά πόσο είναι συμβατά με κάποια από τις μεθόδους παλινδρόμησης.

3.5.1 Παλινδρόμηση κορυφογραμμής

Έχει αποδειχθεί ότι η μέθοδος ελαχίστων τετραγώνων παράγει τη καλύτερη γραμμική προσέγγιση. Υπάρχει, όμως η περίπτωση όπου οι μεταβλητές είναι συσχετισμένες με αποτέλεσμα η ακρίβεια της συγκεκριμένης μεθόδου να μειώνεται δραματικά.

Η παλινδρόμηση κορυφογραμμής (ridge regression) αποτελεί μια ευρέως γνωστή μέθοδο για τη κανονικοποίηση μοντέλων που δεν μπορούν να προσεγγιστούν καλά από τη μέθοδο παλινδρόμησης ελαχίστων τετραγώνων είτε γιατί δεν υπάρχουν κάποιες ανεξάρτητες μεταβλητές είτε γιατί δεν είναι ανεξάρτητες. Αυτή η τεχνική επιτυγχάνει μειώνοντας ουσιαστικά το μεγάλο μέγεθος κάποιων παραμέτρων της παλινδρόμησης.

3.5.2 *Lasso*

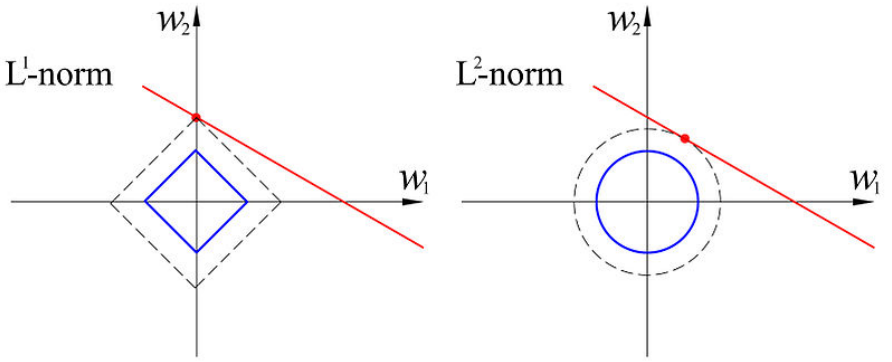
Η μέθοδος Lasso (Least Absolute shrinkage and selection operator) είναι μια κανονικοποιημένη εκδοχή της μεθόδου ελαχίστων τετραγώνων περιορίζοντας όμως το μέγεθος του διανύσματος εισόδου. Στη στατιστική και στην μηχανική μάθηση, η τεχνική Lasso είναι μια μέθοδος παλινδρόμησης και χρησιμοποιείται για επιλογή μεταβλητών στη μοντελοποίηση κάποιου προβλήματος καθώς και για κανονικοποίηση, προκειμένου να ενισχυθεί η ακρίβεια της πρόβλεψης. Η συγκεκριμένη τεχνική διατυπώθηκε αρχικά για τα μοντέλα ελαχίστων τετραγώνων. Αποκαλύπτει σημαντικές πληροφορίες για τη συμπεριφορά του εκτιμητή, συμπεριλαμβανομένης της σχέσης του με τη τεχνική παλινδρόμησης κορυφογραμμών (Ridge Regression) και τη τεχνική καλύτερης επιλογής υποσυνόλου (Best Subset Selection).

Αν και αρχικά είχε οριστεί για τη μέθοδο των ελαχίστων τετραγώνων, η τεχνική κανονικοποίησης Lasso εύκολα επεκτείνεται σε ένα ευρύ φάσμα στατιστικών μοντέλων, συμπεριλαμβανομένων γενικευμένα γραμμικά μοντέλα, γενικευμένες εξισώσεις εκτίμησης, αναλογικά μοντέλα κινδύνων, και M-estimators, με σχετικά εύκολο τρόπο.

Ο Robert Tibshirani εισήγαγε τη τεχνική Lasso για να βελτιώσει την ακρίβεια της πρόβλεψης των μοντέλων παλινδρόμησης με την αλλαγή της διαδικασίας προσαρμογής του μοντέλου, έτσι ώστε να επιλέγετε μόνο ένα υποσύνολο των διανυσμάτων εισόδου για χρήση στο τελικό μοντέλο. [21]

Πριν από τη τεχνική Lasso, η πιο ευρέως χρησιμοποιούμενη μέθοδος για την επιλογή των διανυσμάτων εισόδου που θα συμπεριλαμβάνονταν στο μοντέλο ήταν κλιμακωτή επιλογή, η οποία βελτιώνει μόνο την ακρίβεια της πρόβλεψης, σε ορισμένες περιπτώσεις, όπως για παράδειγμα όταν μόνο λίγα διανύσματα έχουν μια ισχυρή σχέση με το αποτέλεσμα. Ωστόσο, σε άλλες περιπτώσεις, μπορεί να κάνει το σφάλμα της πρόβλεψης μεγαλύτερο. Επίσης, εκείνη την εποχή, η παλινδρόμηση κορυφογραμμών ήταν η πιο δημοφιλής τεχνική για τη βελτίωση της ακρίβειας πρόβλεψης. Η παλινδρόμηση κορυφογραμμών βελτιώνει το σφάλμα πρόβλεψης με τη συρρίκνωση των μεγάλων συντελεστών των διανυσμάτων εισόδου προκειμένου να μειωθεί το πρόβλημα της υπερπροσαρμογής.

Με τη τεχνική Lasso μπορούν να τεθούν κάποιοι από τους συντελεστές στο μηδέν, ενώ με τη τεχνική παλινδρόμησης κορυφογραμμών, η οποία φαίνεται επιφανειακά παρόμοια, δεν μπορούν να τεθούν στο μηδέν. Αυτό



Σχήμα 3.4: L1 και L2 norms

οφείλεται στη διαφορά στο σχήμα των ορίων των περιορισμών στις δύο περιπτώσεις. Τόσο η τεχνική Lasso όσο και η τεχνική παλινδρόμησης κορυφογραμμών μπορούν να ερμηνευθούν ως ελαχιστοποιήσεις της ίδιας συνάρτηση.

$$\min_{\beta_0,\beta}\{\frac{1}{N} \parallel y - \beta_0 - \mathrm{X}\beta \parallel_2^2\}$$

αλλά έχουν τους εξής περιορισμούς:$\parallel \beta \parallel_1 \leqslant t$ για τη τεχική LASSO και $\parallel \beta \parallel_2^2 \leqslant t$ για τη τεχνική παλινδρόμησης κορυφογραμμών. Από το σχήμα, μπορεί κανείς να δει ότι η περιοχή περιορισμού που ορίζεται από την $l1$ νόρμα αποτελεί ένα τετράγωνο που περιστρέφεται έτσι ώστε οι γωνίες να βρίσκονται στους άξονες, ενώ η περιοχή που ορίζεται από τη νόρμα $l2$ είναι ένας κύκλος, η οποία είναι περιστροφικά αμετάβλητη και, ως εκ τούτου, δεν έχει γωνίες.

3.6 Κανονικοποίηση

3.6.1 Αραιές αναπαραστάσεις

Η αρχή της αραιής κωδικοποίησης είναι καίριας σημασίας για πολλούς τομείς των σύγχρονων επιστημών, η απλούστερη εξήγηση σε ένα συγκεκριμένο φαινόμενο πρέπει να προτιμάται σε σχέση με πιο περίπλοκη. Στο πλαίσιο της μηχανικής μάθησης, παίρνει τη μορφή μεταβλητών ή χαρακτηριστικών. Πιο συγκεκριμένα, κάνει το μοντέλο ή τη πρόβλεψη πιο ερμηνεύσιμη ή υπολογιστικά φθηνότερη. Συνήθως, τα μοντέλα που χρησιμοποιούνται στη μηχανική μάθηση είναι τεράστια σε όγκο και πολλές φορές ο υπολογισμός των παραμέτρων του μοντέλου δεν είναι δυνατό να επιλυθεί λόγο υπολογιστικής πολυπλοκότητας.

Κάποιες από τις λύσεις που έχουν προταθεί στη βιβλιογραφία για μείωση του διανύσματος των χαρακτηριστικών, περιλαμβάνουν αυτοματοποιημένους και μη αλγόριθμους για τη διαγραφή χαρακτηριστικών τα οποία θεωρούμε ότι δεν συμβάλουν αρκετά για να υπάρχουν στο μοντέλο. Με

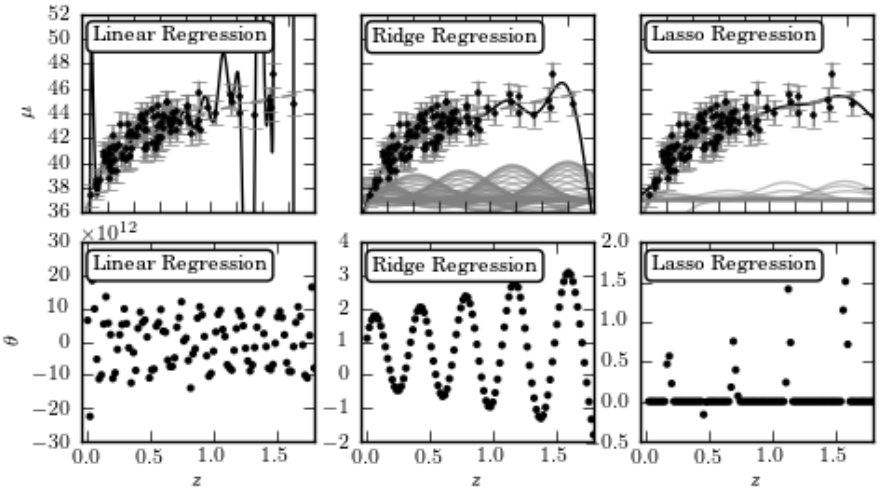
αυτό το τρόπο όμως προφανώς χάνουμε χαρακτηριστικά από το μοντέλο μας. Μια καλύτερη τεχνική για να κρατήσουμε όλα τα χαρακτηριστικά του μοντέλου μας είναι η χρήση κανονικοποίησης. Χρησιμοποιώντας τεχνικές κανονικοποίησης μπορούμε να θέσουμε βάρη σε κάποιες παραμέτρους έτσι ώστε να θεωρούνται πιο σημαντικές από κάποιες άλλες.

Επίσης, η τεχνική της κανονικοποίησης προσπαθεί να επιλύσει ακόμα ένα σημαντικό πρόβλημα. Το πρόβλημα της υπερπροσαρμογής και της υποπροσαρμογής. Η υποπροσαρμογή παρουσιάζεται όταν το μοντέλο που προσπαθούμε να εφαρμόσουμε στο πρόβλημα είναι πολύ απλό με αποτέλεσμα ο αλγόριθμος να αποτυγχάνει. Αντίθετα, το πρόβλημα της υπερπροσαρμογής είναι όταν προσπαθήσουμε να μοντελοποιήσουμε ένα πρόβλημα με όσα περισσότερα χαρακτηριστικά μπορούμε. Αυτό έχει σαν αποτέλεσμα όμως το μοντέλο να ταιριάζει πάρα πολύ καλά στο πρόβλημα άλλα αποτυγχάνει για τη γενική περίπτωση πολύ απλά επειδή είναι πολύ συγκεκριμένο. Συγκεκριμένα προσπαθώντας να μειώσουμε το σφάλμα της συνάρτησης κόστους το μοντέλο γίνεται πολύ συγκεκριμένο και αποτυγχάνει να γενικεύσει καινούργια παραδείγματα.

Ένας από τους πρωτοπόρους στη θεωρία της Κανονικοποίησης ήταν ο Tikhonov ο οποίος πρότεινε το 1963 τη συγκεκριμένη μέθοδο για την επίλυση κακώς ορισμένων (ill-posed) προβλημάτων. Τα προβλήματα (ill-posed) αποτελούν προβλήματα που είτε δεν έχουν λύση, είτε έχουν πολλές, είτε αυτές είναι ευαίσθητες σε σχέση με την επιλογή παραμέτρων. Μέσω της κανονικοποίησης, στοχεύουμε στην αύξηση της ικανότητας γενίκευσης του συστήματος, με κόστος την αύξηση του σφάλματος εκπαίδευσης. Επίσης, τα περισσότερα προβλήματα έχουν απειρία λύσεων και τα βάρη μπορούν να πάρουν αυθαίρετα υψηλές τιμές, κάτι που δεν είναι επιθυμητό. Επομένως, καταφεύγουμε στην κανονικοποίηση για να επιβάλλουμε μικρές τιμές και ομαλότητα στα βάρη.

Για την επιλογή μεταβλητών σε γραμμικά μοντέλα, οι αραιές αναπαραστάσεις μπορούν να επιτευχθούν άμεσα με το να μειώσουμε κάποιες από τις παραμέτρους του διανύσματος βάρους. Ωστόσο, αυτό οδηγεί σε δυσεπίλυτα συνδυαστικά προβλήματα τάξης NP. Μια συνηθισμένη προσέγγιση του προβλήματος είναι η αντικατάσταση των διαστάσεων του συνόλου από την $l1$-νόρμα.

Μετατρέποντας τις αραιές αναπαραστάσεις ως βελτιστοποιήσεις κυρτών συναρτήσεων έχουμε αρκετά πλεονεκτήματα. Συγκεκριμένα, οδηγούμαστε σε αποδοτικούς αλγόριθμους εκτιμήσεων σε αντίθεση με τα δυσεπίλυτα συνδυαστικά προβλήματα που θα αντιμετωπίζαμε. Επίσης, όταν οι αραιές αναπαραστάσεις είναι καλά ορισμένες, η κανονικοποίηση από την $l1$-νόρμα μετατρέπεται σε πρόβλημα υψηλών διαστάσεων, όπου ο αριθμός των μεταβλητών του μοντέλου μπορεί να είναι εκθετικός σε σύγκριση με τον αριθμό των παρατηρήσεων.



Σχήμα 3.5: Σύγκιση ανάμεσα σε τρείς διαφορετικές τεχνικές παλινδρόμησης

Ένας από τους συνηθέστερους τρόπους για τη κανονικοποίηση της συνάρτησης κόστους είναι η προσθήκη της υπερπαραμέτρου λ και του όρου κανονικοποίησης. Προσθέτοντας, αυτά τα δύο στη συνάρτηση ρυθμίζουμε την επιρροή του όρου στη συνολική συνάρτηση κόστους. Με αυτό το τρόπο επιτυγχάνουμε να προτιμώνται μικρές τιμές βαρών και επομένως τιμωρείται η πολυπλοκότητα των απεικονίσεων. Πρέπει να είμαστε ιδιαίτερα προσεχτικοί στην επιλογή της υπερπαραμέτρου λ έτσι ώστε οι κλίσεις σε έναν αλγόριθμο κατάβασης δυναμικού, να προέρχονται από τη συνάρτηση κόστους και όχι από τον όρο κανονικοποίησης. Με άλλα λόγια, πρέπει να βρούμε τη χρυσή τομή μεταξύ της συνάρτησης κόστους και του όρου κανονικοποίησης έτσι ώστε να μοντελοποιήσουμε το πρόβλημα μειώνοντας όσο περισσότερο μπορούμε το πρόβλημα της υπερπροσαρμογής και της υποπροσαρμογής. [20]

Για τις βελτιστοποιήσεις κυρτών συναρτήσεων το πρόβλημα έχει την εξής μορφή:

$$\min_{w \in \Re^p} f(w) + \lambda \Omega(w) \tag{3.12}$$

όπου η $f : \Re^p \to \Re$ είναι κυρτή διαφορίσιμη συνάρτηση και $\Omega : \Re^p \to \Re$ ασυνεχής μη Ευκλείδεια συνάρτηση.

Στη μηχανική μάθηση προβλέπουμε τα αποτελέσματα $y \in Y$ από παρατηρήσεις $x \in X$. Οι παρατηρήσεις αυτές αναπαρίστανται τις πιο πολλές φορές από διανύσματα διαστάσεων p έτσι ώστε το $X = \Re^p$. Για ν ζεύγη δεδομένων $\{(x^{(i)}, y^{(i)}) \in \Re^p \times Y; i = 1, ...n\}$, για γραμμικά μοντέλα έχουμε, $f(w) := \frac{1}{n} \sum_{i=1}^{n} l(y^{(i)}, w^T x^{(i)})$. Στη συνέχεια, κάποια από τα τυπικά μοντέλα για συναρτήσεις απωλειών είναι η μέθοδος των ελάχιστων τετραγώνων, για παράδειγμα $l(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$ όπου $y \in \Re$, και οι λογαριθμικές απώλειες $l(y, \hat{y}) = log(1 + e^{-y\hat{y}})$, όπου $y \in \{-1, 1\}$.[31]

Γνωρίζοντας ότι οι λύσεις του προβλήματος έχουν μερικούς μη μηδενικούς συντελεστές Ω συχνά επιλέγεται να είναι η $l1 - norm$ δηλαδή:

$\Omega(\omega)=\sum_{j=1}^{p}|w_j|$.[31] Αυτό οδηγεί στη τεχνική LASSO. Με το να χρησιμοποιήσουμε τη νόρμα $l1$ στη τεχνική της κανονικοποίησης που εφαρμόζουμε επιτυγχάνουμε αραιές αναπαραστάσεις. Συγκεκριμένα, ανάλογα με το βαθμό της κανονικοποίησης, κάποιος αριθμός μηδενικών συντελεστών στο διάνυσμα αποτελεσμάτων θα έχει τιμή μηδέν.

3.6.2 Προσεγγιστικές μέθοδοι

Οι προσεγγιστικές μέθοδοι χρησιμοποιούνται συγκεκριμένα για τη βελτιστοποίηση μιας συνάρτησης της μορφής (12), δηλαδή, συνάρτηση η οποία μπορεί να γραφτεί ως το άθροισμα μιας γενικής διαφορίσιμης συνάρτησης f, και μιας μη-διαφορίσιμης συνάρτησης $\lambda\Omega$. Έχουν κεντρίσει το ενδιαφέρων στη κοινότητα της μηχανικής μάθησης, κυρίως λόγω των ποσοστών της σύγκλισης τους και την ικανότητά τους να αντιμετωπίζουν δύσκολα μη-λεία κυρτά προβλήματα.

Οι προσεγγιστικές μέθοδοι μπορούν να περιγραφούν ως εξής: σε κάθε επανάληψη η συνάρτηση f γραμμικοποιήται γύρω από το τρέχον σημείο και επιλύεται ένα πρόβλημα της μορφής:

$$\min_{w\in\Re} f(w^t) + \nabla f(w^t)^T(w - w^t) + \lambda\Omega(\omega) + \frac{L}{2} \parallel w - w^t \parallel_2^2 \qquad (3.13)$$

Ο όρος δευτέρου βαθμού, που ονομάζεται και προσεγγιστικός όρος, κρατά το αποτέλεσμα σε μια κοντινή περιοχή της τρέχουσας επανάληψης w^t όπου η συνάρτηση f είναι κοντά σε μια γραμμική προσέγγιση της. Το $L > 0$ αποτελεί παράμετρο, η οποία πρέπει ουσιαστικά να είναι ένα άνω φράγμα για τη σταθερά Λιπσχιτζ της ∇f.

Επομένως, το πρόβλημα παίρνει την εξής μορφή:

$$\min_{e\in\Re^p} \frac{1}{2} \parallel w - (w^t - \frac{1}{L}\nabla f(w^t)) \parallel_2^2 + \frac{\lambda}{L}\Omega(w) \qquad (3.14)$$

ρέπει να σημειωθεί ότι όταν η συνάρτηση Ω δεν υπάρχει τότε η συνάρτηση παίρνει τη μορφή $w^{t+1} \leftarrow w^t - \frac{1}{L}\nabla f(w^t)$. Επιπλέον, εάν η συνάρτηση Ω αποτελεί συνάρτηση δείκτη του συνόλου i_C τότε η επίλυση της συνάρτησης 14 είναι η προβολή της κλίσης στο σύνολο C. Αυτό υποδηλώνει ότι η λύση του προσεγγιστικού προβλήματος παρέχει μια ενδιαφέρουσα γενίκευση της κλίσης, και παρακινεί την εισαγωγή της έννοιας του προσεγγιστικού όρου που σχετίζεται με τον όρο της κανονικοποίησης $\lambda\Omega$.

Ο προσεγγιστικός όρος τον οποίο ορίσαμε ως $Prox_\mu\Omega$, ορίστηκε αρχικά από το Moreau (1962) ως η συνάρτηση που χαρτογραφεί ένα διάνυσμα $u \in \Re^p$ στη μοναδική λύση

$$\min_{w\in\Re^p} \frac{1}{2} \parallel u - w \parallel^2 + \mu\Omega(\omega) \qquad (3.15)$$

Ο βασικός προσεγγιστικός αλγόριθμος χρησιμοποιεί τη λύση του προβλήματος 14 ως την επόμενη επανάληψη w^{t+1}. Ωστόσο, άλλες παραλλαγές του συγκεκριμένου αλγορίθμου, διατηρούν δύο μεταβλητές και τις χρησιμοποιούν έτσι ώστε να συνδυάσουν τη λύση του προβλήματος 14 με πληροφορίες σχετικά με προηγούμενα βήματα. Συνήθως, το άνω φράγμα για τη σταθερά Lipschitz ∇f δεν είναι γνωστό, και ακόμη και αν είναι, είναι συχνά προτιμότερο να ληφθεί μια τοπική εκτίμηση. Μία κατάλληλη τιμή για το L μπορεί να ληφθεί αυξάνοντας επαναληπτικά το L κατά ένα σταθερό παράγοντα έως ότου η συνθήκη

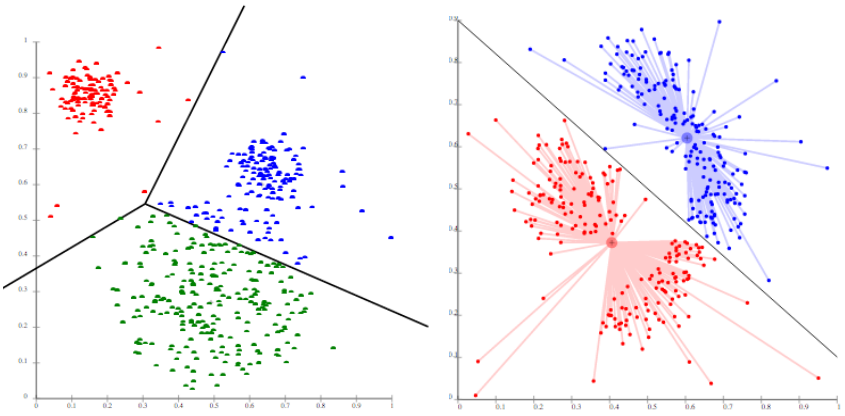
$$f(w_L^*) \leq M_f^L(w^t, w_L^*) := f(w^t) + \nabla f(w^t)^T(w_L^* - w^t) + \frac{L}{2} \parallel w_L^* - w^t \parallel_2^2 \tag{3.16}$$

να ικανοποιηθεί.

3.7 Εναλλακτικές προσεγγιστικές μέθοδοι

3.7.1 *K-Means*

Ο αλγόριθμος K-Means είναι ένας από τους πιο απλούς και δημοφιλέστερους αλγορίθμους συσταδοποίησης (clustering) που ανήκουν στην ευρύτερη κατηγορία των τεχνικών μάθησης χωρίς επίβλεψη.[19] Αποτελεί δημοφιλή αλγόριθμο εξαιτίας της απλότητας της υλοποίησης του και της γραμμικής πολυπλοκότητας του η οποία είναι της τάξης O(n). Ο αλγόριθμος K-Means πρέπει να γνωρίζει εκ των προτέρων των αριθμό K των συστάδων.



Σχήμα 3.6: Αλγόριθμος K-Means

Η κεντρική ιδέα του αλγορίθμου είναι

1. Προσδιορισμός K κέντρων (centroids) – ένα για κάθε συστάδα (cluster)

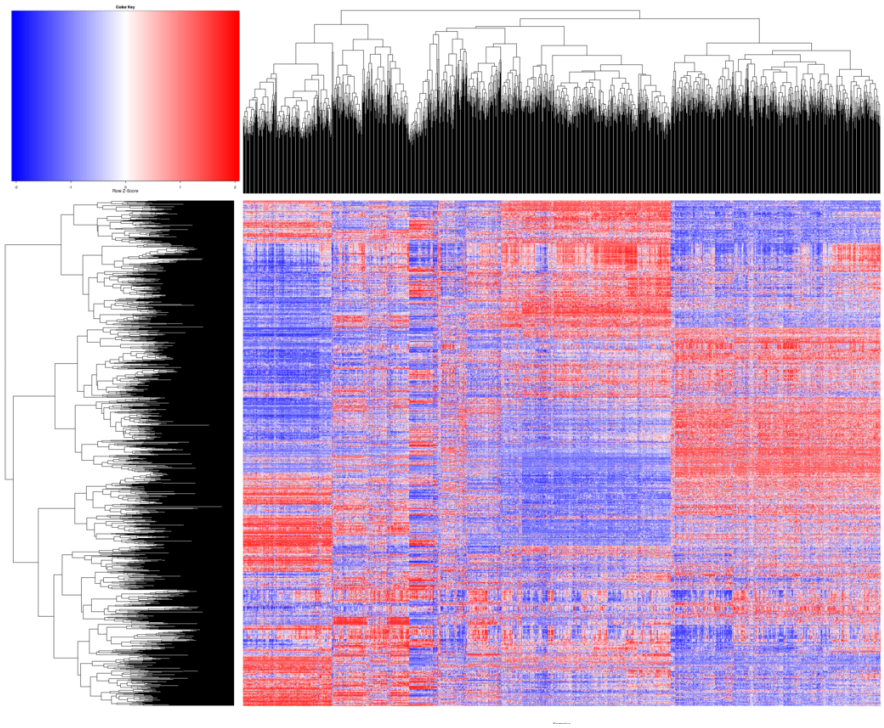
2. Επιλογή κάθε στοιχείου από το σύνολο δεδομένων και συσχέτιση του με το κοντινότερο σε αυτό κέντρο.
3. Υπολογισμός ξανά K νέων κέντρων, τα οποία θα αποτελούν το κέντρο βάρους για κάθε συστάδα που προέκυψε από το προηγούμενο βήμα.
4. Επιλογή κάθε στοιχείου από το σύνολο δεδομένων και συσχέτιση του με το κοντινότερο σε αυτό κέντρο.
5. Επανάληψη των βημάτων 1-4 μέχρι να μην σημειωθούν αντιμεταθέσεις στοιχείων.

Σημείωση: Τα αρχικά κέντρα πρέπει να επιλεγούν με το κατάλληλο τρόπο, διότι για διαφορετικές αρχικές θέσεις έχουμε διαφορετικά αποτελέσματα. Δηλαδή, η αρχική θέση των κέντρων που θα επιλεγούν, επηρεάζει το αποτέλεσμα που θα δώσει ο αλγόριθμος. μια καλή τακτική είναι η επιλογή εκείνων των κέντρων ώστε να απέχουν μεταξύ τους όσο περισσότερο γίνεται.

Έτσι, γίνεται μια επανάληψη της ίδιας διαδικασίας. Αποτέλεσμα αυτής της επανάληψης είναι ότι σε κάθε βήμα τα κέντρα αλλάζουν θέση (ορίζονται νέα) και τα στοιχεία ανατίθενται στη κατάλληλη συστάδα κάθε φορά με βάση το κοντινότερο κέντρο. Όταν σε κάποια επανάληψη δεν σημειωθούν αντιμεταθέσεις στοιχείων, τότε τερματίζει η εκτέλεση του αλγορίθμου. Το αποτέλεσμα που προκύπτει είναι η ομαδοποίηση του συνόλου δεδομένων σε K συστάδες clusters .

3.7.2 Ιεραρχική Ομαδοποίηση

Η ιεραρχικής ομαδοποίησης (hierarchical clustering), βασίζεται στην ιδέα ότι τα αντικείμενα που βρίσκονται πιο κοντά συσχετίζονται περισσότερο σε σχέση αντικείμενα που βρίσκονται πιο μακριά. Ο αλγόριθμος ενώνει αντικείμενα με βάση την απόσταση που απέχουν μεταξύ τους. Κάθε συστάδα (cluster) μπορεί να περιγραφεί σε μεγάλο βαθμό από τη μέγιστη απόσταση που χρειάζεται για να συνδεθούν τα αντικείμενα της συστάδας.[19] Σε διαφορετικές αποστάσεις, δημιουργούνται διαφορετικές συστάδες, η οποίες μπορεί να αναπαρασταθούν χρησιμοποιώντας ένα δενδρόγραμμα, γεγονός που εξηγεί γιατί προέκυψε η ονομασία 'ιεραρχική συσταδοποίηση'. Αυτοί οι αλγόριθμοι δεν παρέχουν μια μοναδική διαμέριση του συνόλου δεδομένων, αλλά παρέχουν μια εκτεταμένη ιεραρχία συστάδων που συγχωνεύονται μεταξύ τους σε συγκεκριμένες αποστάσεις.



Σχήμα 3.7: Ιεραρχική Ομαδοποίηση

Οι αλγόριθμοι ιεραρχικής ομαδοποίησης διαφέρουν μεταξύ τους ανάλογα με το πια μέθοδο χρησιμοποιούν για υπολογισμό των αποστάσεων μεταξύ των αντικειμένων. Εκτός από τις συνηθισμένες συναρτήσεις για υπολογισμό αποστάσεων, ο χρήστης πρέπει επίσης να αποφασίσει πιο κριτήριο σύνδεσης θα χρησιμοποιηθεί. Δημοφιλείς επιλογές είναι γνωστές ως απλή σύνδεση (το ελάχιστο των αποστάσεων αντικειμένου), πλήρη σύνδεση (το μέγιστο των αποστάσεων αντικειμένου) ή UPGMA (αποτελεί το μέσο των αποστάσεων των αντικειμένων). Τέλος, αυτές οι μέθοδοι δεν παράγουν μια μοναδική διαμέριση του συνόλου δεδομένων, αλλά μια ιεραρχία από την οποία εξακολουθεί να χρειάζεται ο χρήστης να επιλέξει τις κατάλληλες συστάδες.

Μέρος IV

ΣΥΝΟΛΟ ΔΕΔΟΜΕΝΩΝ

4

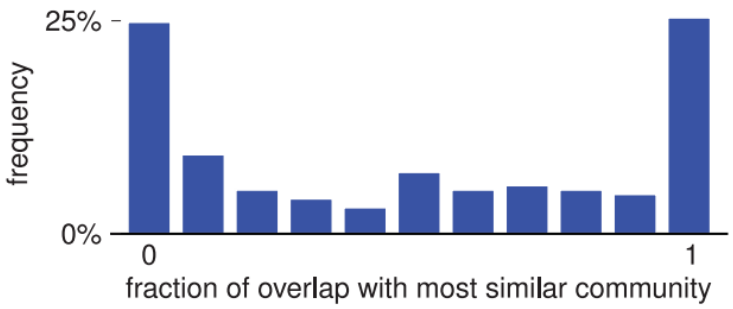
ΠΕΡΙΓΡΑΦΗ ΣΥΝΟΛΟΥ ΔΕΔΟΜΕΝΩΝ

Παρά το γεγονός ότι το σύστημά μας, υλοποιεί αλγόριθμο χωρίς επίβλεψη, θέλουμε να έχουμε για είσοδο επισημασμένα δεδομένα έτσι ώστε να μπορέσουμε να αξιολογήσουμε κατά πόσο πετύχαμε το επιθυμητό αποτέλεσμα. Καταφέραμε να αποκτήσουμε δίκτυα 'φίλων' για τρία από τα μεγαλύτερα κοινωνικά δίκτυα (Facebook, Twitter, Google+).[42][1] [2] [3]

Από το Facebook έχουμε δεδομένα για το προφίλ και τη δομή 10 δικτύων, τα οποία αποτελούνται από 193 κύκλους και 4039 χρήστες. Τα δεδομένα λήφθηκαν χρησιμοποιώντας την εφαρμογή του Facebook. Για την απόκτηση των δεδομένων πραγματοποιήθηκε μια έρευνα μεταξύ 10 χρηστών (κυρίως φοιτητών), οι οποίοι κλήθηκαν να προσδιορίσουν χειροκίνητα σε ποιους κύκλους ανήκει ο κάθε φίλος τους. Η διαδικασία ταξινόμησης των φίλων κάθε χρήστη διήρκησε περίπου 2 έως 3 ώρες για ολόκληρο το δίκτυο τους. Κατά μέσο όρο, οι χρήστες προσδιόρισαν 19 κύκλους στα δίκτυα τους, με μέσο μέγεθος κύκλου 22 φίλων. Παραδείγματα των κύκλων που λήφθηκαν, περιλαμβάνουν φοιτητές, που έχουν κάποια κοινά χαρακτηριστικά μεταξύ τους όπως πανεπιστήμιο, μαθήματα, αθλητικές ομάδες, συγγενείς, και ούτω καθεξής.

Το πιο κάτω σχήμα δείχνει το βαθμό στον οποίο οι 193 (χειροκίνητα επισημασμένοι) κύκλοι, στα 10 δίκτυα που πήραμε από το Facebook, αλληλεπικαλύπτονται (τέμνονται) μεταξύ τους. Περίπου το ένα τέταρτο των επισημασμένων κύκλων είναι ανεξάρτητοι από οποιοδήποτε άλλο κύκλο. Αν και ένα υποσύνολο ίσως να περιέχεται εντός άλλου κύκλου, όπως για παράδειγμα ένα σύνολο φίλων, οι οποίοι είχαν το ίδιο μάθημα, αποτελούν υποσύνολο των φοιτητών που φοιτούν στον ίδιο κλάδο ή πανεπιστήμιο. Το υπόλοιπο 50% των κοινοτήτων που υπάρχουν συμπίπτουν σε κάποιο βαθμό με ένα άλλο κύκλο.

1 https://snap.stanford.edu/data/egonets-Facebook.html
2 http://snap.stanford.edu/data/egonets-Gplus.html
3 http://snap.stanford.edu/data/egonets-Twitter.html



Σχήμα 4.1: Ιστόγραμμα των κύκλων που αλληλεπικαλύπτονται (στο Facebook). Η τιμή μηδέν δείχνει ότι ο κύκλος δεν τέμνεται με κανένα από τους υπόλοιπους κύκλους του χρήστη, ενώ η τιμή ένα υποδεικνύει ότι ένας κύκλος περιλαμβάνεται εντελώς σε ένα άλλο. Περίπου το 25% των κύκλων συμπεριλαμβάνονται εντελώς σε κάποιο άλλο από τους κύκλους.

Για τα άλλα δύο σύνολα δεδομένων, λάβαμε δεδομένα που είναι διαθέσιμα στο κοινό. Από το Google+, λήφθηκαν δεδομένα από 133 δίκτυα, τα οποία αποτελούνται από 479 κύκλους και 106.674 χρήστες. Οι κύκλοι του Google+ είναι εντελώς διαφορετικοί από αυτούς του Facebook, με την έννοια ότι οι χρήστες που τους δημιούργησαν επέλεξαν να τους δημοσιοποιήσουν, και επίσης το Google+ είναι ένα κατευθυνόμενο δίκτυο σε αντίθεση με το Facebook το οποίο δεν είναι (σημειώστε ότι το μοντέλο μας μπορεί να εφαρμοστεί τόσο σε κατευθυνόμενα όσο και σε μη κατευθυνόμενη δίκτυα). Για παράδειγμα, πολλοί από τους κύκλους του Google+ αποτελούν κοινότητες από συγγραφείς, πολιτικούς, ή διασημότητες, οι οποίοι προφανώς δεν είναι στενοί φίλοι αυτών των χρηστών που συνθέτουν κάποιο δίκτυο, αλλά είναι απλά άτομα τα οποία θαυμάζουν αυτούς τους συγκεκριμένους χρήστες και θέλουν να τους ακολουθούν. Τέλος, από το Twitter, λάβαμε δεδομένα από 1.000 δίκτυα, που αποτελούνται από 4.869 κύκλους και 81.362 χρήστες. Τα δίκτυα από το Twitter λήφθηκαν με την εξής τεχνική. Αρχικά επιλέγηκε ένας τυχαίος χρήστης, και με χρήση αλγορίθμου Αναζήτησης Πρώτα σε Βάθος (BFS breadth-first-search) επιλέγονταν χρήστες οι οποίοι είχαν δημιουργήσει τουλάχιστο δύο λίστες με φίλους τους. Η αναζήτηση σταμάτησε μόλις υπήρχαν δεδομένα για 1000 χρήστες. Τα δίκτυα που λήφθηκαν, κυμαίνονται σε μέγεθος από 10 έως 4964 κόμβους.

Υπάρχει η πιθανότητα να υπάρχουν ορισμένες προκαταλήψεις στα δείγματα που έχουμε. Οι χρήστες που έχουμε από το Twitter είναι κυρίως φοιτητές, οι χρήστες του Google+ επέλεξαν να δημοσιοποιήσουν τους κύκλους τους (οι οποίοι δεν περιέχουν προσωπικούς φίλους), και ούτω καθεξής. Για να αντιμετωπιστεί αυτό, Θα ήταν ιδανικό να είχαμε κάποιο μεγαλύτερο σύνολο δεδομένων που θα είχε ως είσοδο το μοντέλο μας, δυστυχώς όμως αυτά τα δεδομένα έχουμε στη διάθεση μας.

Στα δεδομένα μας έχουμε αθροιστικά, 1143 διαφορετικά δίκτυα, 5541 κύκλους και 192075 χρήστες. Οι διαφορές μεγέθους, μεταξύ των συ-

νόλων δεδομένων που έχουμε στη διάθεση μας, αντικατοπτρίζουν απλώς τη διαθεσιμότητα των δεδομένων, από καθεμία από τις τρεις πηγές που μελετήσαμε. Το σύνολο των δεδομένων που έχουμε από το Facebook είναι πλήρως επισημασμένο, με την έννοια ότι έχουμε τις κοινότητες που ο κάθε χρήστης θεωρεί ότι αντικατοπτρίζουν την πραγματικότητα, σε αντίθεση με το Google+ και Twitter για τα οποία έχουμε μόνο δεδομένα τα οποία είναι δημόσια διαθέσιμα.

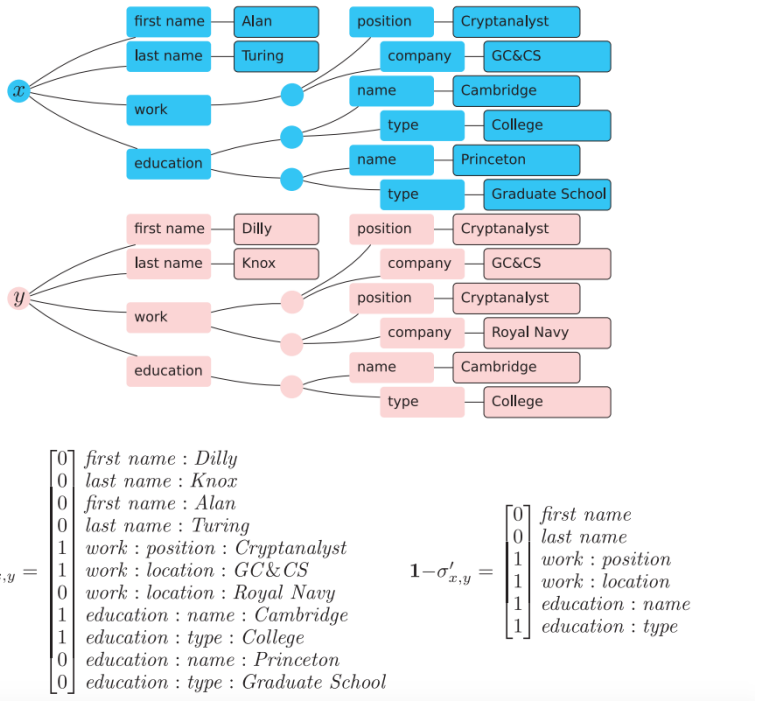
Μέρος V

ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΧΡΗΣΤΩΝ

5

ΚΑΤΑΣΚΕΥΗ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΑΠΟ ΤΑ ΠΡΟΦΙΛ ΤΩΝ ΧΡΗΣΤΩΝ

Οι πληροφορίες για το προφίλ κάποιου χρήστη σε όλα τα σύνολα δεδομένων που έχουμε στη διάθεση μας, θα αναπαρασταθούν σαν ένα δέντρο όπου κάθε επίπεδο κωδικοποιεί όλο και πιο συγκεκριμένες πληροφορίες. με άλλα λόγια, τα προφίλ χρηστών οργανώνονται σε ολοένα και πιο συγκεκριμένες κατηγορίες. Το προφίλ κάποιου χρήστη, περιέχει πληροφορίες για το που εργάζεται ή πιο είναι το επίπεδο σπουδών του, αυτές οι πληροφορίες χωρίζονται σε υποκατηγορίες, για παράδειγμα το όνομα του πανεπιστημίου που φοίτησε ή φοιτά καθώς και το τύπο του πανεπιστημίου. Όπως, παρατηρούμε στο δέντρο (που φαίνεται στο σχήμα 16) τα φύλλα περιέχουν συγκεκριμένες πληροφορίες για αυτές τις υποκατηγορίες.

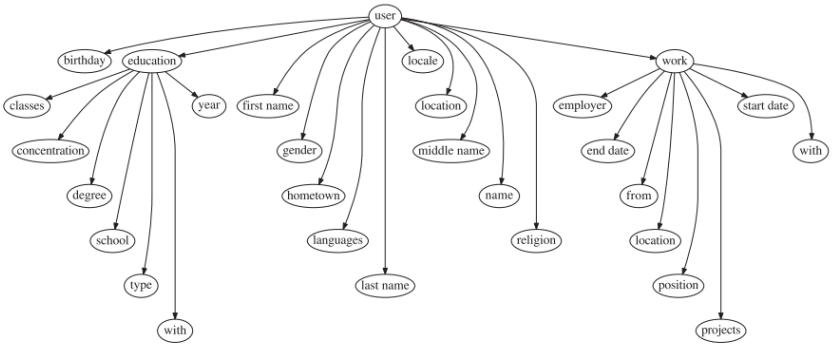


$$\mathbf{1}-\sigma_{x,y}=\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\1\\1\\0\\0\end{bmatrix}\begin{matrix}\textit{first name : Dilly}\\\textit{last name : Knox}\\\textit{first name : Alan}\\\textit{last name : Turing}\\\textit{work : position : Cryptanalyst}\\\textit{work : location : GC\&CS}\\\textit{work : location : Royal Navy}\\\textit{education : name : Cambridge}\\\textit{education : type : College}\\\textit{education : name : Princeton}\\\textit{education : type : Graduate School}\end{matrix}\qquad \mathbf{1}-\sigma'_{x,y}=\begin{bmatrix}0\\0\\1\\1\\1\\1\end{bmatrix}\begin{matrix}\textit{first name}\\\textit{last name}\\\textit{work : position}\\\textit{work : location}\\\textit{education : name}\\\textit{education : type}\end{matrix}$$

Σχήμα 5.1: Κατασκευή χαρακτηριστικών για κάθε προφίλ χρήστη. Τα προφίλ είναι δομημένα σε μορφή δέντρου και κατασκευάζουμε τα χαρακτηριστικά συγκρίνοντας τα μονοπάτια στο δέντρο. Στην εικόνα εμφανίζονται παραδείγματα για δύο χρήστες.

Παρατηρήσαμε ότι οι χρήστες οργανώνουν τους κοινωνικούς τους κύκλους με δύο τρόπους. Είτε σχηματίζουν κύκλους γύρω από τους χρήστες οι οποίοι έχουν κάποια κοινή ιδιότητα, είτε σχηματίζουν κύκλους γύρω από

τους χρήστες οι οποίοι μοιράζονται κάποια κοινή ιδιότητα μαζί τους. Για παράδειγμα, ένας χρήστης μπορεί να έχει αρκετούς φίλους οι οποίοι εργάζονται στη Google αλλά ο ίδιος να μην εργάζεται. Άρα είναι πιθανόν να επιθυμεί να δημιουργήσει κοινωνικούς κύκλους με αυτά τα άτομα αν και δεν έχουν κάποια κοινή ιδιότητα. Σε αυτή τη περίπτωση η κοινή ιδιότητα αυτών των χρηστών, που στο παράδειγμα μας είναι η δουλειά, δεν παίζει καθοριστικό ρόλο στη δημιουργία των κοινωνικών κύκλων του χρήστη. Η δεύτερη περίπτωση, προφανώς, είναι να έχουν κάποια κοινή ιδιότητα, όπως για παράδειγμα να φοιτούν στο ίδιο πανεπιστήμιο, να εργάζονται μαζί ή να κατοικούν στην ίδια περιοχή. Με αυτό το τρόπο, έχουμε τη δυνατότητα να εκτιμήσουμε ποια από αυτές τις υποθέσεις αντιπροσωπεύει καλύτερα τα δεδομένα που λαμβάνουμε.



Σχήμα 5.2: Τα χαρακτηριστικά που πήραμε από το Facebook. Κάθε ένα από τα 26 φύλλα είναι μια κατηγορία από χαρακτηριστικά.

Από το Google+, παίρνουμε δεδομένα από έξι κατηγορίες (φύλο, επώνυμο, θέσεις εργασίας, ιδρύματα, πανεπιστήμια και τόπο διαμονής). Από το Facebook, συλλέξαμε δεδομένα από 26 κατηγορίες, συμπεριλαμβανομένων τόπου διαμονής, ημερομηνία γέννησης, συναδέλφους, καθώς και πολιτικές και θρησκευτικές πεποιθήσεις. Τέλος, από το Twitter παίρνουμε δεδομένα από δύο κατηγορίες, πιο συγκεκριμένα παίρνουμε τα hashtags και τις αναφορές που χρησιμοποίησε κάθε χρήστης σε διάρκεια δύο εβδομάδων. Όπως έχουμε προαναφέρει, στη δομή δεδομένων που θα οργανώσουμε τις πληροφορίες που συλλέξαμε οι κατηγορίες αποτελούν τους γονείς και οι πληροφορίες κάθε κατηγορίας τα φύλλα. Το πλήρες σύνολο στοιχείων που παίρνουμε από το Facebook φαίνεται στο Σχήμα 17.

Θέλουμε να δημιουργήσουμε ένα διάνυσμα διαφορών έτσι ώστε να κωδικοποιήσουμε τις σχέσεις μεταξύ των χρηστών. Ουσιαστικά, θέλουμε να κωδικοποιήσουν τις διαστάσεις του διανύσματος, για να μπορούμε να ξεχωρίζουμε πια χαρακτηριστικά έχουν κοινά δύο χρήστες. Ας υποθέσουμε κάθε χρήστης $v \in V$ έχει ένα προφίλ T_v το οποίο παρουσιάζεται σαν μια δομή δέντρου και $l \in T_v$ είναι ένα φύλλο σε αυτό το δέντρο.[42] Ορίζουμε τη διανυσματική διαφορά $\sigma_{x,y}$ μεταξύ δύο χρηστών x και y ως δυαδικό δείχτη δείχνοντας με αυτό το τρόπο ποια χαρακτηριστικά των δύο χρηστών είναι διαφορετικά (Σχήμα 16, κάτω αριστερά).

$$\sigma_{x,y}[l] = \delta((l \in T_x) \neq (l \in T_y) \tag{5.1}$$

Παρά το γεγονός, ότι, το διάνυσμα διαφορών έχει το πλεονέκτημα ότι κωδικοποιεί τις πληροφορίες του προφίλ με εξαιρετική αναλυτικότητα, παρουσιάζει το μειονέκτημα ότι έχει αρκετά μεγάλες διαστάσεις. (μέχρι 4122 διαστάσεις στα δεδομένα που έχουμε στη διάθεση μας). Ένας τρόπος για να αντιμετωπιστεί αυτό, είναι να δημιουργήσουμε διανύσματα διαφοράς, που βασίζονται στους γονείς των κόμβων φύλλων. με αυτόν τον τρόπο, έχουμε κωδικοποιήσει τις κοινές κατηγορίες που έχουν οι δύο χρήστες, αγνοώντας όμως συγκεκριμένες τιμές με αποτέλεσμα το παραγόμενο διάνυσμα να έχει λιγότερες διαστάσεις (Σχήμα 16 κάτω δεξιά). Όπως φαίνεται στο σχήμα γνωρίζουμε ποια χαρακτηριστικά έχουν κοινά δύο χρήστες αλλά το διάνυσμα δεν περιέχει τις τιμές για αυτά τα χαρακτηριστικά.

$$\sigma'_{x,y}[p] = \sum_{l \in children(p)} \sigma_{x,y}[l] \tag{5.2}$$

Η προσέγγιση αυτή, έχει το πλεονέκτημα, ότι, απαιτεί ένα σταθερό αριθμό διαστάσεων για τα διανύσματα, ανεξάρτητα από το μέγεθος του δικτύου που μελετούμε (26 για το Facebook, 6 για το Google+, 2 για το Twitter).

Με βάση τα διανύσματα διαφοράς $\sigma_{x,y}$ και $\sigma'_{x,y}$, περιγράφουμε πως θα κατασκευαστούν τα χαρακτηριστικά των ακμών $\varphi(x,y)$. Η πρώτη ιδιότητα που πρέπει να μοντελοποιηθεί είναι ότι τα μέλη των κύκλων πρέπει να έχουν κοινές σχέσεις μεταξύ τους.

$$\varphi^1(x,y) = (1; -\sigma_{x,y}) \tag{5.3}$$

Η δεύτερη ιδιότητα που θέλουμε να μοντελοποιήσουμε είναι, τα μέλη των κύκλων να έχουν κοινές σχέσεις με το χρήστη στον οποίο ανήκει το δίκτυο. Σε αυτή την περίπτωση τα χαρακτηριστικά ορίζονται με βάση το χρήστη υ στον οποίο ανήκει το δίκτυο. Έχουμε:

$$\varphi^2(x,y) = (1; -|\sigma_{x,u} - \sigma_{y,u}|) \tag{5.4}$$

Χρησιμοποιώντας τις δύο πιο πάνω ιδιότητες έχουμε την δυνατότητα να αξιολογήσουμε ποιος μηχανισμός ταιριάζει καλύτερα για κάθε χρήστη, διότι κάθε χρήστης έχει ελαφρώς διαφορετική αντίληψη για το ποιους χρήστες πρέπει να περιέχουν οι κύκλοι του. Και στις δύο περιπτώσεις, υπάρχει ένα σταθερό χαρακτηριστικό ('1'), το οποίο μετρά το βαθμό στον οποίο οι κύκλοι αποτελούνται από φίλους. Αυτό μας επιτρέπει να προβλέψουμε τα μέλη ενός κύκλου, ακόμη και για χρήστες που δεν έχουν τα απαραίτητα στοιχεία στο προφίλ τους, μόνο από το να γνωρίζουμε με ποια μέλη συνδέονται.

Επίσης, για τη μορφή του διανύσματος $\sigma'_{x,y}$ με τις λιγότερες διαστάσεις έχουμε:

$$\psi^1(x,y) = (1; \sigma'_{x,y}),\ \psi^2(x,y) = (1; -|\sigma'_{x,u} - \sigma'_{y,u}|) \tag{5.5}$$

Τέλος, προσδιορίσαμε τέσσερις διαφορετικούς τρόπους για να παρουσιάσουμε τη συμβατότητα μεταξύ δύο χρηστών. Επιλέξαμε δύο τρόπους για

τη κατασκευή των διανυσματικών διαφορών $(\sigma_{x,y}, \sigma'_{x,y})$, και δύο τρόπους $(\varphi(x,y), \psi(x,y))$ για να καταγράψουμε τη συμβατότητα μεταξύ δύο χρηστών. Τα χαρακτηριστικά αυτά, είναι σχεδιασμένα για τη μοντελοποίηση των δύο ακόλουθων συμπεριφορών που προαναφέραμε:

1. Οι χρήστες στους οποίους ανήκει το δίκτυο, κατασκευάσουν κύκλους γύρω από κοινές σχέσεις που έχουν μεταξύ τους οι φίλοι τους.
2. Οι χρήστες στους οποίους ανήκει το δίκτυο, κατασκευάσουν κύκλους γύρω από κοινές σχέσεις που έχουν μεταξύ τους οι φίλοι τους και σχέσεις που έχουν και αυτοί με τους φίλους τους.

Μέρος VI

ΥΛΟΠΟΙΗΣΗ

6

ΚΩΔΙΚΑΣ

6.1 *Egonetworks.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package egonetworks;

import solver.Egosolver;
import solver.Unknowns;

/**
 *
 * @author orbit
 */
public class Egonetworks {

  /**
   * @param args
   *            the command line arguments
   */
  public static void main(String[] args) {

    String edgeFile = "";
    String featureFile = "";
    String fileName =  "src/Data/facebook";

```

```
    edgeFile = fileName + "/3980.edgesm";
    featureFile = fileName + "/3980.featmDeletedSorted";

    FeatureGraph fg = Egoreader.userFeatureGraphReader(edgeFile,
        featureFile);

    LogLikelihood ll = new LogLikelihood(fg);


    Unknowns uknowns = Egosolver.solve(ll, 100, 10, 1e-6);

    boolean[][] circles = uknowns.getCircles();

    int circleNo = 0;
    for (boolean[] circle : circles) {
      System.out.println("Circle is " + circleNo + " and contains
          :");
      circleNo++;
      for (int node = 0; node < circle.length; node++) {
        if (circle[node]) {
          System.out.print(" " + node);
        }
      }
      System.out.println("");
    }

  }

}
```

6.2 *Egoreader.java*

```
/*
 * To change this license header, choose License Headers in
     Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package egonetworks;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.awt.Point;
import java.util.*;

/**
 *
 * @author orbit
 */
```

```
public class Egoreader {

    public static List<String[]> readStrippedFields(String
        filename) {
22      BufferedReader br = null;
        List<String[]> combs = new ArrayList<>();
        try {
            br = new BufferedReader(new FileReader(filename));
            boolean finished = false;
27          do {
                String line = br.readLine();
                if ((line != null) && !line.isEmpty()) {
                    String[] linefields = line.split("\\s+");
                    combs.add(linefields);
32              } else {
                    finished = true;
                }
            } while (!finished);
        } catch (FileNotFoundException e) {
37          e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (br != null) {
42              try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
47          }
        }
        return combs;
    }

52  public static boolean[][] cycleReader(int nodes, String
        cyclesFileName) {
        List<boolean[]> combs = new ArrayList<>();
        List<String[]> separatedfields = Egoreader.
            readStrippedFields(cyclesFileName);
        for (String[] linefields : separatedfields) {
            boolean[] cycle = new boolean[nodes];
57          for (String field : linefields) {
                int node = Integer.parseInt(field);
                cycle[node] = true;
            }
            combs.add(cycle);
62      }
        boolean[][] cycles = new boolean[combs.size()][];
        int index = 0;
        for (boolean[] cycle : combs) {
            cycles[index++] = cycle;
67      }
        return cycles;
```

```
    }

    public static boolean[][] edgeReader(String edgeFileName) {
        Set<Point> edgepoints = new HashSet<Point>();
        int upper = Integer.MIN_VALUE;
        int lower = Integer.MAX_VALUE;
        List<String[]> separatedfields = Egoreader.
            readStrippedFields(edgeFileName);
        for (String[] endpoints : separatedfields) {
            int from = Integer.parseInt(endpoints[0]);
            int to = Integer.parseInt(endpoints[1]);
            if (from > upper) {
                upper = from;
            }
            if (to > upper) {
                upper = to;
            }

            if (from < lower) {
                lower = from;
            }
            if (to < lower) {
                lower = to;
            }
            edgepoints.add(new Point(from, to));
        }

        if ((upper == Integer.MIN_VALUE) || (lower == Integer.
            MAX_VALUE)) {
            return null;
        }

        if (lower != 1) {
            return null;
        }

        if (upper <= 0) {
            return null;
        }

        int nodes = upper + 1;

        boolean[][] edges = new boolean[nodes][nodes];

        for (Point p : edgepoints) {
            edges[p.x][p.y] = true;
        }

        return edges;
    }

    public static double[][] featureReader(int nodes, String
        featureFileName) {
```

```
        double[][] features = new double[nodes][];
        List<String[]> separatedfields = Egoreader.
            readStrippedFields(featureFileName);
        for (String[] linefields : separatedfields) {
            int node = Integer.parseInt(linefields[0]);
            double[] sfeats = new double[linefields.length - 1];
            for (int i = 0; i < linefields.length - 1; i++) {
                sfeats[0] = Double.parseDouble(linefields[i + 1])
                    ;
            }
            features[node] = sfeats;
        }
        if (features[0].length == 0) {
            return null;
        }

        int checkpoint = features[0].length;
        for (double[] feature : features) {
            if ((feature == null) || (feature.length !=
                checkpoint)) {
                return null;
            }
        }
        return features;
    }

    public static FeatureGraph userFeatureGraphReader(String
        edgeFileName, String featureFileName) {
        boolean[][] edges = Egoreader.edgeReader(edgeFileName);
        int nodes = edges.length;
        double[][] features = Egoreader.featureReader(nodes,
            featureFileName);
        int checkpoint = features[0].length;
        double[][][] pairwisefeatures = new double[nodes][nodes][
            checkpoint];
        for (int i = 0; i < nodes; i++) {
            for (int j = 0; j < nodes; j++) {
                for (int l = 0; l < checkpoint; l++) {
                    pairwisefeatures[i][j][l] = features[i][l] ==
                        features[j][l] ? 1 : 0;
                }
            }
        }
        return new FeatureGraph(edges, pairwisefeatures);
    }
}
```

6.3 *FeatureGraph.java*

```
/*
```

```
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package egonetworks;

/**
 *
 * @author orbit
 */
public class FeatureGraph {

    public boolean[][] graph;
    public double[][][] features;
    int numedges;

    public FeatureGraph(boolean[][] edges, double[][][] features)
         {
        this.features = features;
        this.numedges = 0;
        this.graph = edges;
        for (int x = 0; x < this.graph.length; x++) {
            for (int y = 0; y < this.graph.length; y++) {
                if (this.graph[x][y]) {
                    this.numedges++;
                }
            }
        }
    }

    public int getNodes() {
        return this.graph.length;
    }

    public int getFeatureDim() {
        return this.graph.length;
    }
```

```
48
    public boolean hasEdge(int x, int y) {
        return graph[x][y];
    }

53  public double[][][] getFeatures() {
        return features;
    }

    public int getNumedges() {
58      return numedges;
    }

    public double[] getFeatures(int x, int y) {
        return this.features[x][y];
63  }
}
```

6.4 *LogLikelihood.java*

```
1  /*
       egocirlces is free software: you can redistribute it and/or
           modify
       it under the terms of the GNU General Public License as
           published by
       the Free Software Foundation, either version 3 of the License
           , or
       (at your option) any later version.
6
       Sevirus is distributed in the hope that it will be useful,
       but WITHOUT ANY WARRANTY; without even the implied warranty
           of
       MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
       GNU General Public License for more details.
11
       You should have received a copy of the GNU General Public
           License
       along with Foobar.  If not, see <http://www.gnu.org/licenses
           />.

    */
16  package egonetworks;


    import solver.Unknowns;
    import java.math.*;
21  import qpbosolver.*;
    import java.util.*;
    import java.util.function.*;

    /**
```

```
 *
 * @author orbit
 */
public class LogLikelihood {

    public FeatureGraph g;

    public LogLikelihood(FeatureGraph g) {
        this.g = g;
    }

    public double getL(Unknowns uknowns) {
        int nodes = this.g.getNodes();
        double result = Math.sqrt(2 * uknowns.getK());
        result *= (nodes * nodes + this.g.getNumedges());

        double maximum = 0;
        for (int x = 0; x < nodes; x++) {
            for (int y = 0; y < nodes; y++) {
                double temp = 0;
                for (double d : this.g.getFeatures(x, y)) {
                    temp += (d * d);
                }
                temp = Math.sqrt(temp);

                if (maximum < temp) {
                    maximum = temp;
                }
            }
        }

        return result * maximum;

    }

    public double Phi(int x, int y, Unknowns uknowns) {
        double phi = 0;
        double[] features = this.g.getFeatures(x, y);
        for (int k = 0; k < uknowns.getK(); k++) {
            double temp = 0;
            double[] tau = uknowns.getTau()[k];
            for (int l = 0; l < features.length; l++) {
                temp += features[l] * tau[l];
            }
            phi += temp;
        }

        for (int k = 0; k < uknowns.getK(); k++) {
            if (uknowns.getCircles()[k][x] && uknowns.getCircles
                ()[k][y]) {
                double temp = 0;
                double[] tau = uknowns.getTau()[k];
                double[] theta = uknowns.getTheta()[k];
```

```
                for (int l = 0; l < features.length; l++) {
                    temp += features[l] * (theta[l] - tau[l]);
                }
                phi += temp;
            }
        }
        return phi;
    }

    public double NakedLogLikelihood(int x, int y, Unknowns
        uknowns) {
        if (x == y) {
            return 0;
        }
        double term1 = 0;
        double term2 = 0;
        double phi = this.Phi(x, y, uknowns);
        if (this.g.hasEdge(x, y)) {
            term1 = phi;
        }
        term2 = Math.log(1 + Math.exp(phi));
        return -term1 + term2;
    }

    public double NakedLogLikelihood(Unknowns uknowns) {
        double sum = 0;
        int nodes = this.g.getNodes();
        for (int x = 0; x < nodes; x++) {
            for (int y = 0; y < nodes; y++) {
                sum += this.NakedLogLikelihood(x, y, uknowns);
            }
        }

        return sum;
    }

    public double[][] derivativeofTheta(Unknowns uknowns){
        int nodes = this.g.getNodes();
        int featurenum=this.g.getFeatureDim();

        double [][] params=new double[nodes][featurenum];

        for (int k = 0; k < uknowns.getK(); k++) {
            for(int l=0;l<featurenum;l++){
                for (int x = 0; x < nodes; x++) {
                for (int y = 0; y < nodes; y++) {
                    double weight=Math.exp(this.Phi(x, y, uknowns
                        ));
                    weight = -( weight / (1+weight));
                    if(g.hasEdge(x, y)){
                        weight += 1;
                    }
```

```
                    if (uknowns.getCircles()[k][x] && uknowns.
                        getCircles()[k][y]){
                        params[k][l] += weight * this.g.
                            getFeatures()[x][y][l];
                    } else {
                        params[k][l] += 0;
                    }
                }
            }
            }

        }

        return params;

    }

    public double[][] derivativeofTau(Unknowns uknowns){
        int nodes = this.g.getNodes();
        int featurenum=this.g.getFeatureDim();

        double [][] params=new double[nodes][featurenum];

        for (int k = 0; k < uknowns.getK(); k++) {
            for(int l=0;l<featurenum;l++){
                for (int x = 0; x < nodes; x++) {
                for (int y = 0; y < nodes; y++) {
                    double weight=Math.exp(this.Phi(x, y, uknowns
                        ));
                    weight = -( weight / (1+weight));
                    if(g.hasEdge(x, y)){
                        weight += 1;
                    }
                    if (uknowns.getCircles()[k][x] && uknowns.
                        getCircles()[k][y]){
                        params[k][l] += 0;
                    } else {
                        params[k][l] += weight * this.g.
                            getFeatures()[x][y][l];
                    }
                }
            }
            }

        }

        return params;

    }

    public double bic(Unknowns uknowns) {
        int nodes = this.g.getNodes();
        int K = uknowns.getK();
```

```
176         //return (-2 * this.NakedLogLikelihood() + this.graph.
                length * this.uknowns.circles.length * Math.log(this.
                numedges));
            return (-2 * this.NakedLogLikelihood(uknowns) + nodes * K
                 * Math.log(nodes * (nodes - 1)));
        }

    }
```

6.5 *Validator.java*

```
    /*
        egocirlces is free software: you can redistribute it and/or
            modify
        it under the terms of the GNU General Public License as
            published by
        the Free Software Foundation, either version 3 of the License
            , or
5       (at your option) any later version.

        Sevirus is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty
            of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
10      GNU General Public License for more details.

        You should have received a copy of the GNU General Public
            License
        along with Foobar.  If not, see <http://www.gnu.org/licenses
            />.


15
     */
    package egonetworks;

    /**
20   *
     * @author orbit
     */
    public class Validator {

25      public static double getBER(boolean[] circle1, boolean[]
            circle2) {
            int a1 = 0;
            int a2 = 0;
            int count1 = 0;
            int count2 = 0;
30

            for (int x = 0; x < circle1.length; x++) {
                if (circle1[x]) {
                    if (!circle2[x]) {
```

```
                    a1++;
                }
                count1++;
            } else {
                if (circle2[x]) {
                    a2++;
                }
                count2++;
            }
        }

        double b1 = a1;
        double b2 = a2;

        return 0.5 * ((b1 / (count1 + 1)) + (b2 / (count2 + 1)));
    }

    public static double evaluate(boolean[][] circles1, boolean
        [][] circles2) {
        double err = 0;
        for (int f = 0; f < circles1.length; f++) {
            double minimum = Validator.getBER(circles1[f],
                circles2[0]);
            for (int g = 1; g < circles2.length; g++) {
                double temp = Validator.getBER(circles1[f],
                    circles2[g]);
                if (temp < minimum) {
                    minimum = temp;
                }
            }
            err += minimum;
        }

        return (err / circles1.length);
    }
}
```

6.6 *PairwiseEnergy.java*

```
/*
 * To change this license header, choose License Headers in
    Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package qpbosolver;
import java.util.*;

/**
 *
 * @author orbit
```

```
 */
public class PairwiseEnergy{
    public int x,y;
    public double  e_0_0,e_0_1,e_1_0,e_1_1;
}
```

6.7 *QPBOInput.java*

```
/*
 * To change this license header, choose License Headers in
     Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package qpbosolver;
import java.util.*;

/**
 *
 * @author orbit
 */
public class QPBOInput{
    public int nodes;
    public int edges;
    public PairwiseEnergy[] pairwiseenergies;
    public UnaryEnergy[]   unaryenergies;
}
```

6.8 *QPBOSolution.java*

```
/*
 * To change this license header, choose License Headers in
     Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package qpbosolver;

/**
 *
 * @author orbit
 */
public class QPBOSolution {

     static {
       if (System.getProperty("os.name").equals("Linux")) {
         System.load("/home/piyiotisk/Downloads/qpbo/jnicpbo/
             Release/libjnicpbo.so");
```

```
    }else{
      System.loadLibrary("libjnicpbo.dll");
    }
   }

    public native void create_qpbo(int nodes,int edges);
    public native void add_unary_energies(int node,double e_0,
        double e_1);
    public native void add_pairwise_energies(int x,int y,double
        e_0_0,double e_0_1,double e_1_0,double e_1_1);
    public native void find_solution();
    public native int get_flag(int node);
    public native void dispose_qpbo();

}
```

6.9 *Qpbosolver.java*

```
/*
 * To change this license header, choose License Headers in
     Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package qpbosolver;
/**
 *
 * @author orbit
 */
public class Qpbosolver {


    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        QPBOSolution solution = new  QPBOSolution();

        QPBOInput input = new QPBOInput();
        input.nodes=2;
        input.edges=1;
        UnaryEnergy u1=new UnaryEnergy();
        u1.e_0=0;
        u1.e_1=2;
        UnaryEnergy u2=new UnaryEnergy();
        u1.e_0=3;
        u1.e_1=6;
        PairwiseEnergy p1=new PairwiseEnergy();
        p1.x=0;
        p1.y=1;
        p1.e_0_0=2;
```

```
        p1.e_0_1=3;
        p1.e_1_0=4;
        p1.e_1_1=6;

        input.unaryenergies=new UnaryEnergy[]{u1,u2};
        input.pairwiseenergies=new PairwiseEnergy[]{p1};
        int[] flags=new int[input.nodes];

        solution.create_qpbo(input.nodes,input.edges);
        int index=0;
        for(UnaryEnergy ue : input.unaryenergies){
            solution.add_unary_energies(index++,ue.e_0,ue.e_1);
        }

        for(PairwiseEnergy pe : input.pairwiseenergies){
            solution.add_pairwise_energies(pe.x,pe.y,pe.e_0_0,pe.
                e_0_1,pe.e_1_0,pe.e_1_1);
        }

        solution.find_solution();


        for(int i=0;i<input.nodes;i++){
            flags[i]=solution.get_flag(i);
            System.out.println("flag "+Integer.toString(i)+" has
                value "+flags[i]);
        }

        solution.dispose_qpbo();
    }
}
```

6.10 *UnaryEnergy.java*

```
/*
 * To change this license header, choose License Headers in
     Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package qpbosolver;
import java.util.*;

/**
 *
 * @author orbit
 */
public class UnaryEnergy {
    public double  e_0,e_1;
}
```

6.11 *CircleSolver.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package solver;

import egonetworks.LogLikelihood;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import qpbosolver.PairwiseEnergy;
import qpbosolver.QPBOInput;
import qpbosolver.QPBOSolution;
import qpbosolver.UnaryEnergy;

/**
 *
 * @author orbit
 */
public class CircleSolver {

  LogLikelihood ll;
  Random rr;

  public CircleSolver(LogLikelihood ll) {
    this.ll = ll;
    this.rr = new Random();
  }

  public Unknowns findCircles(Unknowns unknowns) {
    unknowns = unknowns.modifiableCircles();
    int K = unknowns.getK();
```

```
    for (int k = 0; k < K; k++) {
      boolean[] temp = this.solveQPBO(k, unknowns);
      unknowns.circles[k] = temp;
    }
    return unknowns;
  }

  public QPBOInput FormulateQPBO(int circle, Unknowns unknowns) {
    int nodes = this.ll.g.getNodes();
    unknowns = unknowns.modifiableCircles();
    List<PairwiseEnergy> lista = new ArrayList<>();

    for (int x = 0; x < nodes; x++) {
      for (int y = 0; y < nodes; y++) {
        PairwiseEnergy temp = new PairwiseEnergy();
        temp.x = x;
        temp.y = y;
        unknowns.getCircles()[circle][x] = false;
        unknowns.getCircles()[circle][y] = false;
        temp.e_0_0 = ll.NakedLogLikelihood(x, y, unknowns);
        unknowns.getCircles()[circle][x] = false;
        unknowns.getCircles()[circle][y] = true;
        temp.e_0_1 = ll.NakedLogLikelihood(x, y, unknowns);
        unknowns.getCircles()[circle][x] = true;
        unknowns.getCircles()[circle][y] = false;
        temp.e_1_0 = ll.NakedLogLikelihood(x, y, unknowns);
        unknowns.getCircles()[circle][x] = true;
        unknowns.getCircles()[circle][y] = true;
        temp.e_1_1 = ll.NakedLogLikelihood(x, y, unknowns);
        lista.add(temp);
      }
    }

    QPBOInput input = new QPBOInput();
    input.nodes = nodes;
    input.edges = nodes * (nodes - 1);
    input.unaryenergies = new UnaryEnergy[0];
    input.pairwiseenergies = lista.toArray(new PairwiseEnergy[0])
        ;
    return input;
  }

  public boolean[] solveQPBO(int circle, Unknowns unknowns) {
    QPBOSolution solution = new QPBOSolution();
    QPBOInput input = this.FormulateQPBO(circle, unknowns);
    solution.create_qpbo(input.nodes, input.edges);
    boolean[] flags = new boolean[input.nodes];
    int index = 0;
    for (UnaryEnergy ue : input.unaryenergies) {
      solution.add_unary_energies(index++, ue.e_0, ue.e_1);
    }

    for (PairwiseEnergy pe : input.pairwiseenergies) {
```

```
      if (pe.x != pe.y) {
        solution.add_pairwise_energies(pe.x, pe.y, pe.e_0_0, pe.
            e_0_1, pe.e_1_0, pe.e_1_1);
      }
    }

    solution.find_solution();

    for (int i = 0; i < input.nodes; i++) {
      int temp = solution.get_flag(i);
      System.out.println("flag " + Integer.toString(i) + " has
          value " + temp);

      if (temp == 0) {
        flags[i] = false;
      } else if (temp == 1) {
        flags[i] = true;
      } else {
        flags[i] = rr.nextBoolean();
      }
    }

    solution.dispose_qpbo();
    return flags;
  }

}
```

6.12 *Egosolver.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
```

```
package solver;

import egonetworks.LogLikelihood;
import java.math.*;

/**
 *
 * @author orbit
 */
public class Egosolver {

    public static Unknowns solve(LogLikelihood ll,double lambda,
        int K, double tolerance) {
            Unknowns unknowns = Unknowns.generateInitial(lambda,K
                ,ll.g.getNodes(),ll.g.getFeatureDim());
            CircleSolver cslv=new CircleSolver(ll);
            double previous_value=ll.NakedLogLikelihood(unknowns)
                 + unknowns.getRegularization(lambda);;
            double err=0;
            do {
                Unknowns u1=cslv.findCircles(unknowns);
                Unknowns u2=ProximalSolver.solve(u1, ll,lambda);
                double after_value=ll.NakedLogLikelihood(u2) + u2
                    .getRegularization(lambda);
                System.out.println("after="+after_value+",
                    previous="+previous_value);
                err=Math.abs(after_value-previous_value);
                previous_value=after_value;
                unknowns=u2;
            }while(err >= tolerance);
            return unknowns;
        }


}
```

6.13 *ProximalMethod.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package solver;


import java.util.function.*;
import java.math.*;

/**
 *
 * @author orbit
 */
public class ProximalMethod {

    Function<double[], Double> objective;
    Function<double[], double[]> derivative;
    double lambda;
    double L;

    ProximalMethod(Function<double[], Double> objective, Function
        <double[], double[]> derivative, double lambda, double L)
         {
        this.lambda = lambda;
        this.L = L;
        this.objective = objective;
        this.derivative = derivative;
    }

    public double[] proximal_projection(double mu, double[] x) {
        double[] result = (double[]) x.clone();
        for (int i = 0; i < x.length; i++) {

            if (Math.abs(x[i]) > mu) {
                result[i] = x[i] - mu * Math.signum(x[i]);
            } else {
                result[i] = 0;
            }
        }
        return result;
    }

    public double[] solve(double[] initial, double tolerance) {
        double[] x = (double[]) initial.clone();
        double err = 0;
        do {
            double[] direction = (double[]) x.clone();
            double[] temp = this.derivative.apply(x);
```

```
            for (int i = 0; i < x.length; i++) {
                direction[i] = x[i] - temp[i] / this.L;
            }
            double[] next_x = this.proximal_projection(this.
                lambda / this.L, direction);
            err = 0;
            for (int i = 0; i < x.length; i++) {
                err += ((next_x[i] - x[i]) * (next_x[i] - x[i]));
            }
            err = Math.sqrt(err);
        } while (err >= tolerance);

        System.out.println("heee");
        //debug
        for (int i = 0; i < x.length; i++) {
      System.out.println(x);
    }

        return x;
    }
}
```

6.14 *ProximalSolver.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package solver;

import egonetworks.FeatureGraph;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
```

```
import qpbosolver.PairwiseEnergy;
import qpbosolver.QPBOInput;
import qpbosolver.UnaryEnergy;
import egonetworks.*;
import java.util.function.*;
import qpbosolver.QPBOSolution;

/**
 *
 * @author orbit
 */
public class ProximalSolver {

    public static class Objective implements Function<double[],
        Double> {

        public Double apply(double[] x) {
            return ll.NakedLogLikelihood(unknowns.unzip(x)) +
                unknowns.getRegularization(this.lambda);
        }

        Unknowns unknowns;
        LogLikelihood ll;
        double lambda;

        public Objective(Unknowns unknowns, LogLikelihood ll,
            double lambda) {
            this.unknowns = unknowns;
            this.ll = ll;
            this.lambda=lambda;
        }
    }

    public static class Derivative implements Function<double[],
        double[]> {

        public double[] apply(double[] loc) {

          //debug
          for (int i = 0; i < loc.length; i++) {
            System.out.println(loc[i]);
      }


            Unknowns temp=unknowns.unzip(loc);
            double[][] dertheta = ll.derivativeofTheta(temp);
            double[][] dertau = ll.derivativeofTau(temp);

            double[] params = new double[2 * dertheta.length *
                dertheta[0].length];
            int index = 0;
            for (int k = 0; k < dertheta.length; k++) {
                for (int x = 0; x < dertheta[k].length; x++) {
```

```
                    params[index++] = dertheta[k][x];
                }
            }
            for (int k = 0; k < dertheta.length; k++) {
                for (int x = 0; x < dertheta[k].length; x++) {
                    params[index++] = dertau[k][x];
                }
            }
            return params;
        }

        Unknowns unknowns;
        LogLikelihood ll;

        public Derivative(Unknowns unknowns, LogLikelihood ll) {
            this.unknowns = unknowns;
            this.ll = ll;
        }
    }


    public static Unknowns solve(Unknowns unknowns, LogLikelihood
         ll,double lambda){
        double L=ll.getL(unknowns);
        Derivative derivative=new Derivative(unknowns,ll);
        Objective objective=new Objective(unknowns,ll,lambda);
        ProximalMethod pxm = new ProximalMethod(objective,
            derivative,lambda,L);
        double [] ending=pxm.solve(unknowns.makeVector(), 0.01);
        return unknowns.unzip(ending);
    }

}
```

6.15 *Unknowns.java*

```
/*
    egocirlces is free software: you can redistribute it and/or
        modify
    it under the terms of the GNU General Public License as
        published by
    the Free Software Foundation, either version 3 of the License
        , or
    (at your option) any later version.

    Sevirus is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty
        of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.
```

```
    You should have received a copy of the GNU General Public
        License
    along with Foobar.  If not, see <http://www.gnu.org/licenses
        />.

 */
package solver;

import java.util.Random;

/**
 *
 * @author orbit
 */
public class Unknowns {

    boolean[][] circles;
    double[][] tau;
    double[][] theta;

    private Unknowns(){}

    public double getRegularization(double lambda) {
        double temp = 0;
        for (int k = 0; k < this.tau.length; k++) {
            for (int l = 0; l < this.tau[k].length; l++) {
                temp += Math.abs(this.theta[k][l]);
                temp += Math.abs(this.tau[k][l]);
            }
        }

        return lambda * temp;
    }

    public int getK() {
        return circles.length;
    }

    public boolean[][] getCircles() {
        return circles;
    }

    public double[][] getTau() {
        return tau;
    }

    public double[][] getTheta() {
        return theta;
    }


    public static Unknowns generateInitial(double lambda, int K,
        int nodes, int features) {
```

```
        //initialize cycle data randomly
        Unknowns unknowns = new Unknowns();
        unknowns.tau = new double[K][features];
        unknowns.theta = new double[K][features];
        unknowns.circles = new boolean[K][nodes];
        Random rr = new Random();
        for (int x = 0; x < nodes; x++) {
            for (int k = 0; k < K; k++) {
                unknowns.circles[k][x] = rr.nextBoolean();
            }
        }
        return unknowns;
    }

    public Unknowns unzip(double [] params){

        Unknowns unknowns=new Unknowns();
        unknowns.circles=this.circles;
        int index=0;

        for(int k=0;k<theta.length;k++){
            for(int x=0;x<theta[k].length;x++){
                unknowns.theta[k][x]=params[index++];
            }
        }

        for(int k=0;k<theta.length;k++){
            for(int x=0;x<theta[k].length;x++){
                unknowns.tau[k][x]=params[index++];
            }
        }
        return unknowns;
    }

    public double[] makeVector(){
        int index=0;

        double [] params=new double[2*theta.length*theta[0].
            length];

        for(int k=0;k<theta.length;k++){
            for(int x=0;x<theta[k].length;x++){
                params[index++]=this.theta[k][x];
            }
        }

        for(int k=0;k<theta.length;k++){
            for(int x=0;x<theta[k].length;x++){
                params[index++]=this.tau[k][x];
            }
        }

        //debug
```

```
        for (int i = 0; i < params.length; i++) {
      System.out.println(params[i]);
    }

        return params;
    }

    public Unknowns modifiableCircles(){

        Unknowns unknowns=new Unknowns();
        unknowns.tau=this.tau;
        unknowns.theta=this.theta;

        int K = this.circles.length;
        int nodes=this.circles[0].length;

        boolean[][] tempcircles = new boolean[K][nodes];
        for (int k = 0; k < K; k++) {
            tempcircles[k] = (boolean[]) this.circles[k].clone();
        }

        unknowns.circles=tempcircles;
        return unknowns;
    }

}
```

BIBLIOGRAPHY

[1] Freebase. URL www.freebase.com/m/06h0c8. (Cited on page 9.)

[2] What is network density – and how do you calculate it? URL http://www.the-vital-edge.com/what-is-network-density/. (Cited on page 22.)

[3] Network science. URL https://en.wikipedia.org/wiki/Network_science. (Cited on pages 21, 22, and 23.)

[4] Graph theory. URL http://webwhompers.com/graph-theory.html. (Cited on page 22.)

[5] Clustering coefficient. URL https://en.wikipedia.org/wiki/Clustering_coefficient. (Cited on page 23.)

[6] Wikipedia facebook. URL https://el.wikipedia.org/wiki/Facebook. (Cited on page 10.)

[7] Wikipedia twitter. URL https://el.wikipedia.org/wiki/twitter. (Cited on page 11.)

[8] Social network statistics. URL http://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/. (Cited on pages 10 and 12.)

[9] Wikipedia google+. URL https://el.wikipedia.org/wiki/Google. (Cited on page 11.)

[10] Social networking. URL www.whatis.techtarget.com/definition/social-networking. (Cited on page 9.)

[11] Scienece of social blog. URL https://community.lithium.com/t5/Science-of-Social-blog/Social-Network-Analysis-101/ba-p/5706. (Cited on page 21.)

[12] Social graph: Concepts and issues. URL http://readwrite.com/2007/09/12/social_graph_concepts_and_issues. (Cited on page 18.)

[13] www.ll.mit.edu. URL www.ll.mit.edu/publications/journal/pdf/vol20_no1/20_1_5_Campbell.pdf. (Cited on page 15.)

[14] Defining the clustering coefficient. URL www.networkscience.wordpress.com. (Cited on page 26.)

[15] Connectedness and components. URL www.fna.fi/blog/2012/11/07/tutorial-5-connectedness-and-components. (Cited on page 26.)

[16] Node centrality (graph theory). URL https://en.wikipedia.org/wiki/Network_science. (Cited on page 27.)

[17] Machine learning wikipedia. URL https://www.wikipedia.org. (Cited on page 31.)

[18] Supervised learning wikipedia. URL https://en.wikipedia.org/wiki/Supervised_learning. (Cited on page 33.)

[19] Cluster analysis. URL https://en.wikipedia.org/wiki/Cluster_analysis. (Cited on pages 50 and 51.)

[20] Regularization (mathematics). URL https://en.wikipedia.org/wiki/Regularization_(mathematics). (Cited on page 48.)

[21] Lasso. URL https://en.wikipedia.org/wiki/Lasso_(statistics). (Cited on page 45.)

[22] Facebook: No. 1 globally. *BusinessWeek*, 2010. (Cited on page 18.)

[23] Is google's social graph api a creeping privacy violation? *ReadWriteWeb*, 2010. (Cited on page 18.)

[24] Renewing old resolutions for the new year. *Official Google Blog*, 2012. (Cited on page 18.)

[25] Mislove A. Inferring user profiles in online social networks. in international conference on web search and data mining. 2010. (Cited on page 5.)

[26] Tsiara Ageliki. Sort images with random forests. Master's thesis, University of Ioannina, January 2012. (Cited on page 32.)

[27] Airoldi, S Blei, Fienberg, and Xing E. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 2008. (Cited on page 42.)

[28] H CHUN and H KWAK. Comparison of online social relations in volume vs interaction. 2008. (Cited on page 14.)

[29] Alessia D'Andrea. *An Overview of Methods for Virtual Social Network Analysis*. Number 978-1-84882-228-3. Springer, 2009. (Cited on page 15.)

[30] Easley David and Kleinberg Jon. *Networks Crowds and Markets: Reasoning about a Highly Connected World.* Cambridge University Press, 2010. URL https://www.cs.cornell.edu/home/kleinber/networks-book/networks-book-ch02. (Cited on pages 16 and 17.)

[31] Bach Francis, Jenatton Rodolphe, Mairal Julien, and Obozinski Guillaume. *Convex Optimization with Sparsity-Inducing Norms*. (Cited on pages 48 and 49.)

[32] Lawyer Glenn. Understanding the influence of all nodes in a network. *Nature*, 2015. (Cited on page 27.)

[33] InformationWeek. Google launches social graph api. 2010. (Cited on page 18.)

[34] M Ivaldi. Academy spinoff to database for complex network analysis. an innovative approach to a new technology. 2012. (Cited on page 15.)

[35] Tamminen Janne, Lee Kung-Chung, and Piche Robert. Graph theory, 2013. (Cited on page 16.)

[36] Scott John. *Social Network Analysis*. 2012. URL https://books.google.gr/books?id=MJoIGBfYDGEC&pg=PA164&lpg=PA164&dq=circles+and+characteristics+of+graph+social+network&source=bl&ots=zwBy_ZYo5f&sig=5nGPjoHvUpsYDiPCIDX4H57gVJs&hl=en&sa=X&ved=0CEYQ6AEwBmoVChMIrrDs3ozqyAIVBRYsCh0dJgC2#v=onepage&q&f=false. (Cited on page 16.)

[37] Michael Jordan. *Computer Science Handbook*. Chapman and Hall, 2004. (Cited on page 34.)

[38] Lancichinetti and Fortunato. Social circles. 2009. (Cited on page 4.)

[39] Efraim Turban Linda Lai. Groups formation and operations in the web 2.0 environment and social networks. *Springer*, 17, September 2008. (Cited on page 13.)

[40] D. MacKay. Information theory, inference and learning algorithms. *Cambridge University Press*, 2003. (Cited on page 39.)

[41] Prodromos Malakasiotis. Identifying parts of speech in greek texts with active learning techniques. Master's thesis, Economical University of Athens, June 2005. (Cited on page 31.)

[42] JULIAN MCAULEY and JURE LESKOVEC. *Discovering Social Circles in Ego Networks*. PhD thesis, Computer Science Department, Stanford University, February 2014. (Cited on pages 4, 55, and 62.)

[43] M. Newman. Modularity and community structure in networks. *National Academy of Sciences.*, 2006. (Cited on page 5.)

[44] CBS News. Facebook unveils platform for developers of social applications. 2010. (Cited on page 17.)

[45] Evelien Otte and Ronald Rousseau. Social network analysis: a powerful strategy, also for the information sciences. *Journal of Information Science 28*, 2002. (Cited on page 15.)

[46] Carlos Pinheiro. *Social Network Analysis in Telecommunications*. Number 978-1-118-01094-5. John Wiley and Sons, 2011. (Cited on page 15.)

[47] Ravasz and Barabasi. Hierarchical organization in complex networks. 2003. (Cited on page 4.)

[48] M VALAFAR, R REJAIE, and WILLINGER. A study of user interac- tions in flickr. In *In Proceedings of the ACM Workshop on Online Networks*, 2009. (Cited on page 14.)

[49] B VISWANATH, A MISLOVE, M CHA, and K GUMMADI. On the evolution of user interaction in facebook. In *In Proceedings of the ACM Workshop on Online Networks*, 2009. (Cited on page 14.)

[50] C WILSON, B BOE, A SALA, K PUTTASWAMY, and ZHAO. User interactions in social networks and their implications. In *European Conference on Computer Systems*, 2009. (Cited on page 15.)

[51] ZDNet. Facebook's zuckerberg uncorks the social graph. Technical report, ZDNet, 2010. (Cited on page 18.)